无感 FOC 低压吊扇调试手册

三相电机控制 MCU
FU6832S

峰岹科技(深圳)股份有限公司

目录

目录 2
1 概述 4
2 硬件原理与参数配置 5
2.1 硬件原理 5
2.1.1 电源部分 6
2.1.2 芯片主体 6
2.1.3 433 射频接受部分 7
2.1.4 功率驱动部分 7
2.1.5 运放配置电路 8
2.1.6 母线电压采样 8
3 软件原理 9
3.1 电机状态机流程图 9
3.2 程序流程图 11
3.3 程序说明 11
3.3.1 Main 函数: 11
3.3.2 1ms 定时中断 11
3.3.3 FOC 中断 12
3.3.4 CMP3 中断 12
3.3.5 Timer3 中断 12
4 调试步骤 13
4.1 配置电机参数 13
4.1.1 电机参数 13
4.1.2 电机参数测量方法 13
4.1.3 对应程序 14
4.2 确认芯片内部相关数据配置 14
4.3 确认硬件参数 14
4.4 保护参数设置 15
4.5 启动参数配置 16
4.6 硬件驱动电路检测 18
4.7 调试速度环 18
4.8 调试功率环 19
4.8.1 保护可靠性 21

4.8.2 启动稳定性 21
4.8.3 调速方式选择射频调速 21
5 功能介绍 22
5.1 启动调试 22
5.1.1 Omega 启动 22
5.1.2 PLL_Start 启动 23
5.1.3 启动常见问题&解决方式 23
5.2 保护介绍 24
5.2.1 过流保护 24
5.2.2 电压保护 24
5.2.3 缺相保护 25
5.2.4 堵转保护 25
5.2.5 其他保护 26
6 其他常见功能调试 26
6.1 限功率功能 26
6.2 限速功能 26
7 修改记录 26
8 版权说明 27

1 概述

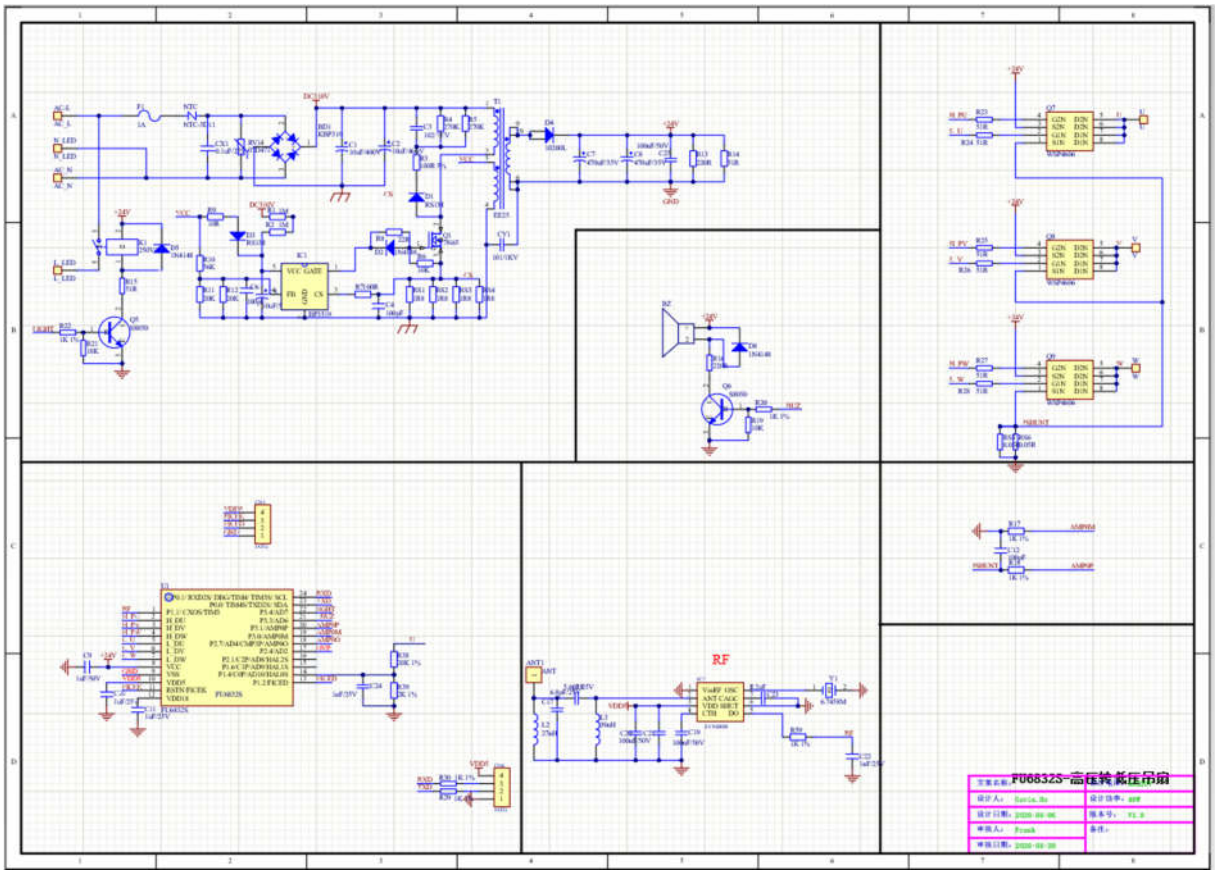
本调试手册详细介绍了如何使用峰岹科技的 FU6832S 芯片，在低压吊扇专用 DEMO 板子上，对直流无刷低压吊扇电机进行无感 FOC 驱动控制。阅读手册时，第二章节硬件原理跟第三章节软件原理可以大致先浏览一遍，重点放在第四章调试步骤。

涉及的软/硬件

软/硬件模块	名称	章节	备注
软件	FU6832S 低压吊扇应用方案标准程序	全部	调试需在该工程软件上进行
硬件	FU6832S-GYZDY-DS_V1.0	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



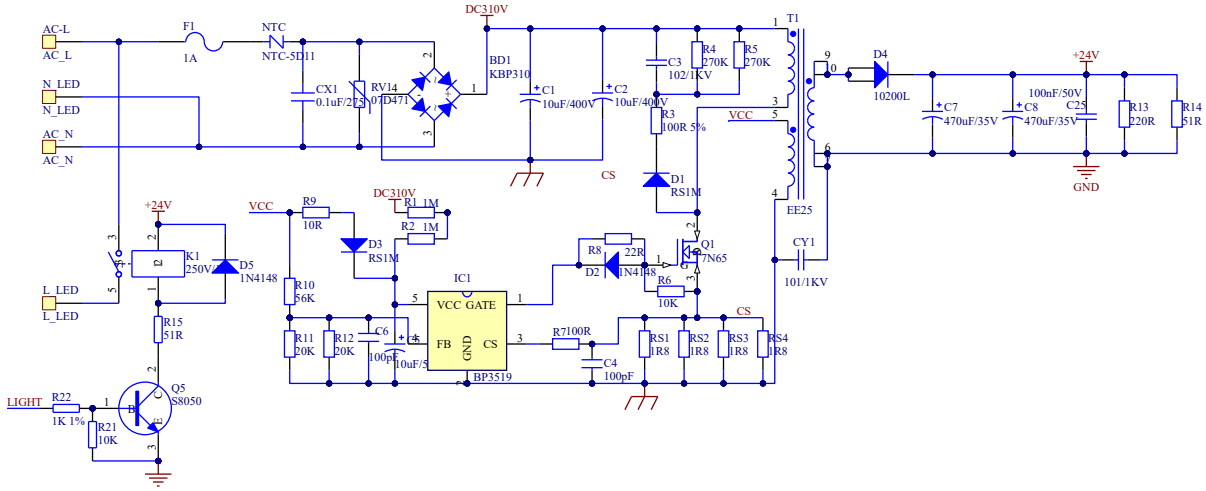
使用方式:

该板子为低压吊扇应用方案的专用 DEMO 板子，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。

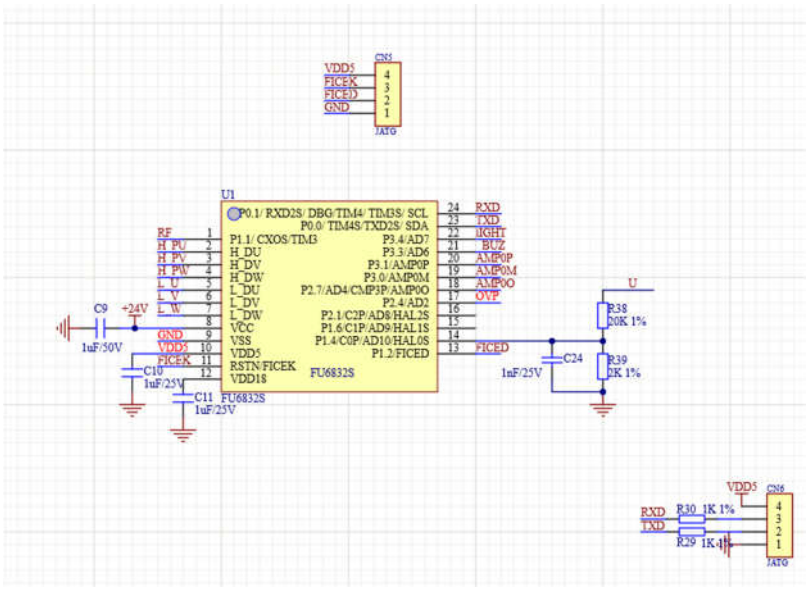
2.1.1 电源部分



使用方式:

交流电直接接 AC-L、AC-N。

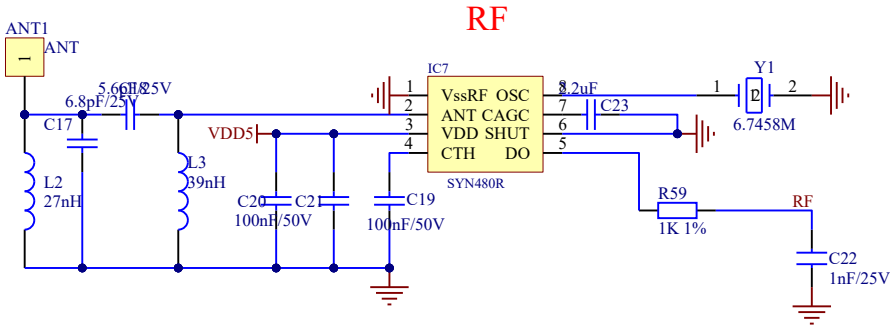
2.1.2 芯片主体



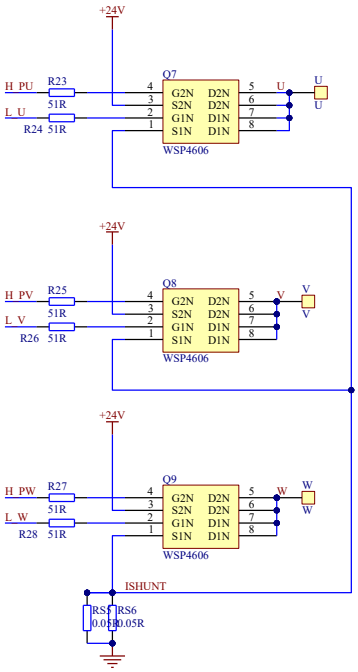
使用方式:

FU6832S 应用于低压 P+NMOSFET 驱动应用。其中 CN5 为烧录线接口，CN6 为串口。

2.1.3 433 射频接受部分



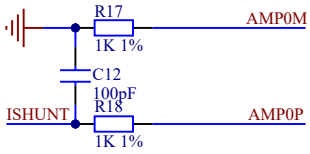
2.1.4 功率驱动部分



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%。

2.1.5 运放配置电路



注意事项:

1. C12 参数不可调整，精度要求 10%；

2.1.6 母线电压采样

注意事项:

1. 用的是内部集成电路，ADC14 采样；

3 软件原理

3.1 电机状态机流程图

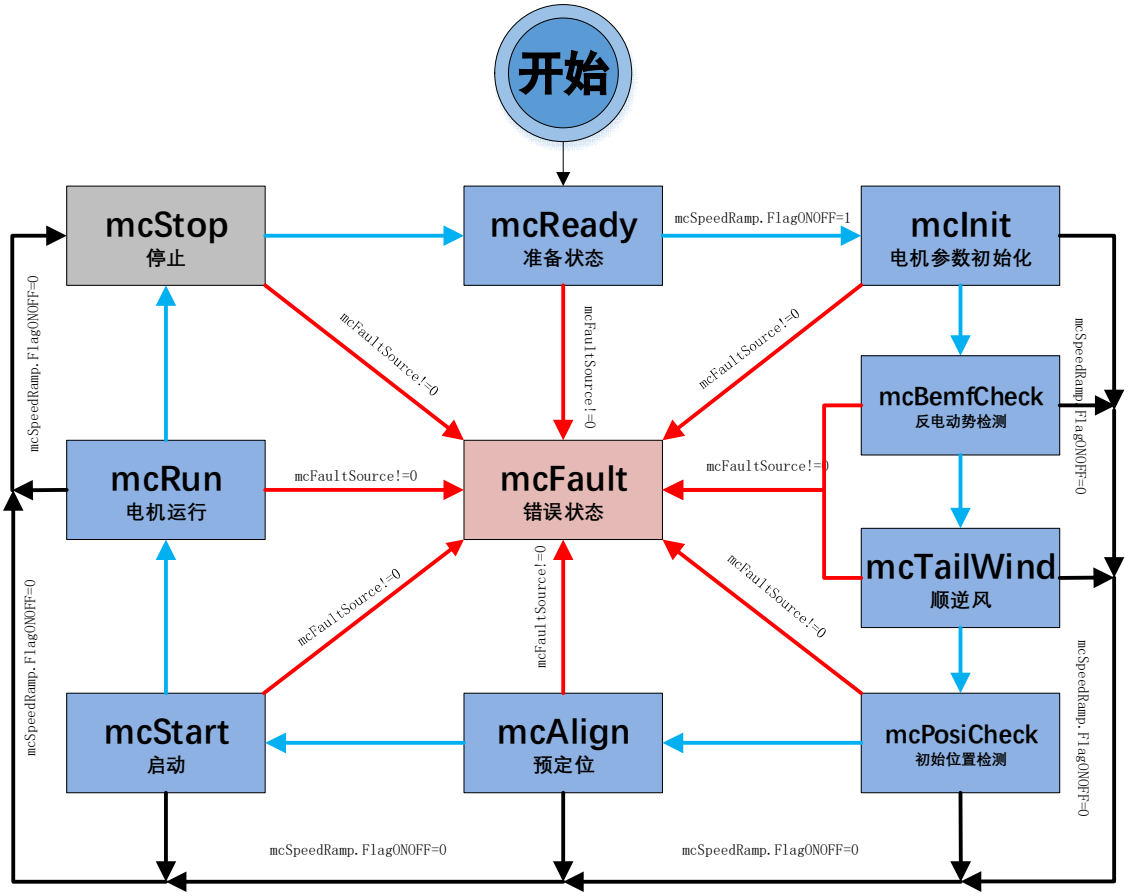


图 3-1 电机状态机流程图

如图所示，电机状态机分为三条路径:

1. 运行: mcReady -> mcInit ->mcBemfCheck-> mcTailWind -> mcPosiCheck -> mcAlign -> mcStart -> mcRun；
2. 停机: 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. mcBemfCheck 、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；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，高速进行刹车降速，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 8 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，mcPosiCheck 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: mcPosiCheck 没使能，，mcTailWind 直接跳转到 mcStart 状态。

3.2 程序流程图

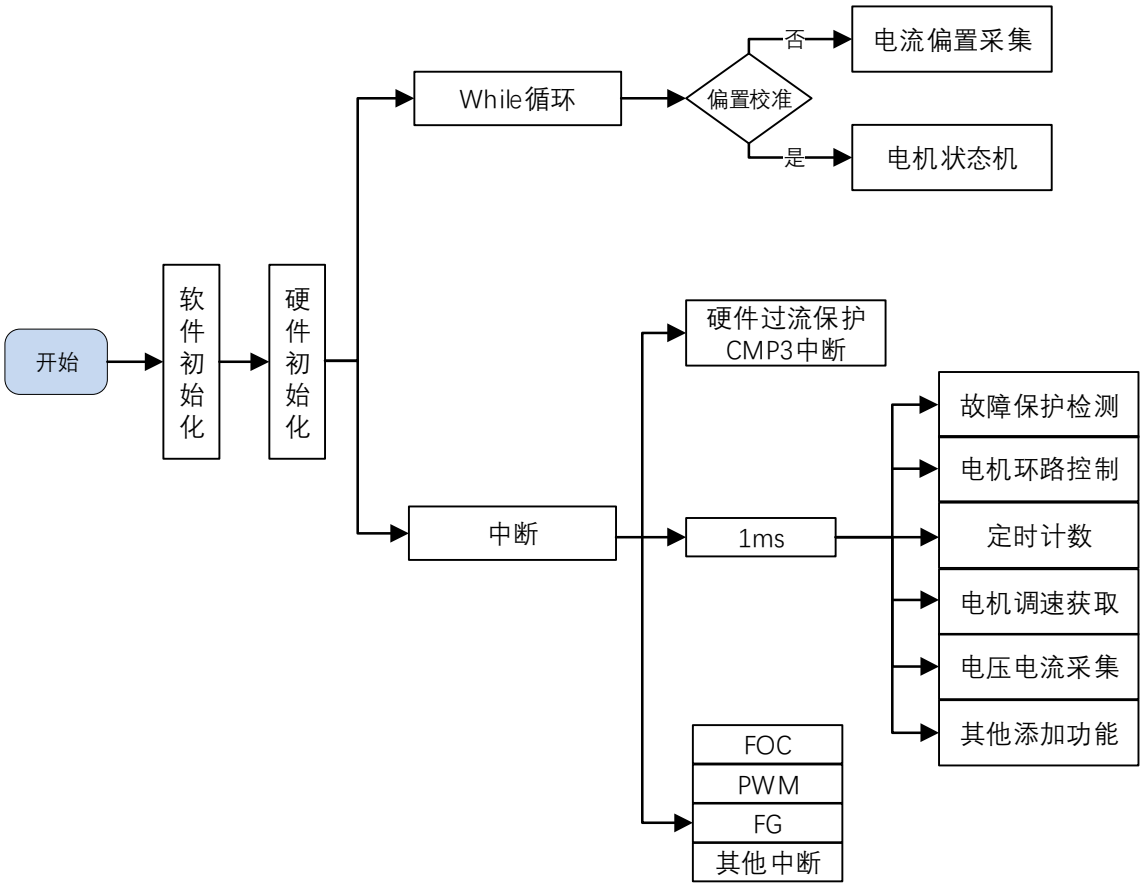


图 3-2 程序执行流程图

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()。

3.3.2 1ms 定时中断

1ms 中断产生标志位 g_1mTick，并在 main 函数调用 TickCycle_1ms 函数。

TickCycle_1ms 函数主要处理调速、故障保护检测、母线电流、母线电压采集等功能：

```
LPF_Func();  //采样及滤波相关处理函数
IR_ContolFunc (); //遥控、蜂鸣器相关处理函数
Limit_Func(); //速度和功率限制函数
```

```
/* -----环路响应处理函数----- */
Speed_response();  //环路响应，如速度环、转矩环、功率环等
Fault_Detection(); //故障保护函数功能，如过欠压保护、启动保护、缺相、堵转等
CurrentZeroDetect(); //刹车电流归零检测
Base_1ms_TIM(); //1ms 相关时间计算处理
StarRampDealwith(); //电机启动爬坡函数处理
FOCTailWindTimeLimit(); // FOC 检测时间
ONOFF_Starttest(); //启停测试
```

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、软件 PLL 启动算法等。

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

Tim3 中断，遥控解码处理。

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 Vpp 和频率 f。计算公式如下:

$$Ke = 1000 * P * \frac{Vpp}{2 * 1.732 * 60 * f}$$

其中，P 为电机极对数。

示例，测量反电动势波形如下:

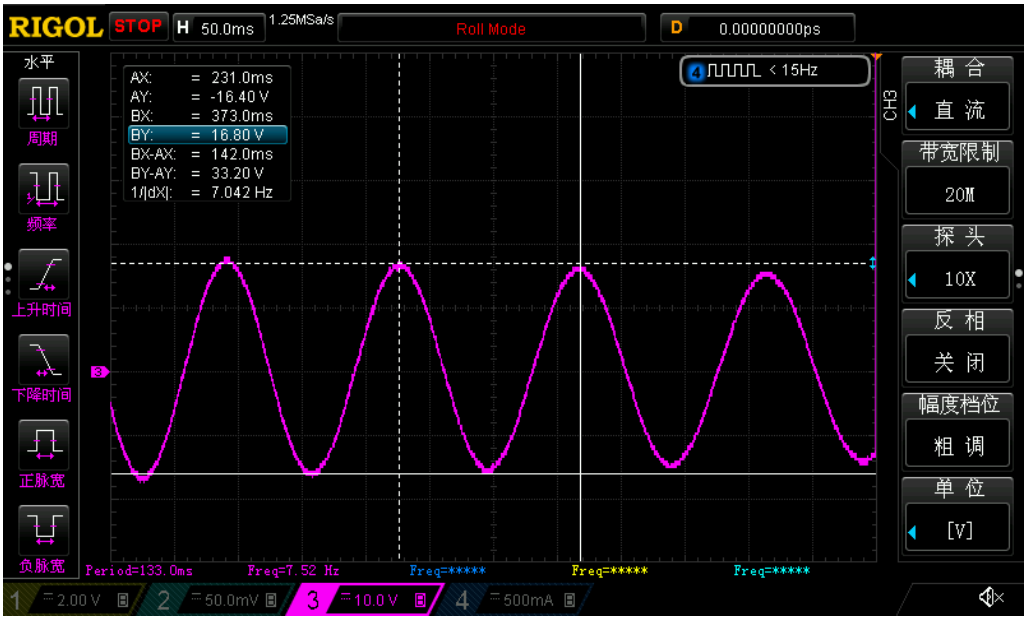


图 4-1 反电动势波形

测量峰峰值 Vpp 为 33.2V，频率 f 为 7.042Hz，极对数 P 为 4，则:

反电动势$Ke = 1000 * 4 * \frac{33.2}{2*1.732*60*7.042} = 90.73$

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

```
/* ---------------------------------------------------------------------------------------------------------------
                                        2. Motor Parameter
-------------------------------------------------------------------------------------------------------------- */
#define Pole_Pairs                  (8.0)               ///< Pole pairs
#define LD                          (0.003)             ///< (H) d-Axis inductance
#define LQ                          (0.003)             ///< (H) q-Axis inductance
#define RS                          (1.0)               ///< (Ω) Phase resistance
#define Ke                          (11.8)              ///< (V/kRPM) Back EMF constant
#define MOTOR_SPEED_BASE            (1000.0)            ///< (RPM) Motor speed base


/* ---------------------------------------------------------------------------------------------------------------
```

4.2 确认芯片内部相关数据配置

```
/* ---------------------------------------------------------------------------------------------------------------
                                        1. Chip parameters
-------------------------------------------------------------------------------------------------------------- */
#define PWM_FREQUENCY               (16.0)              ///< (kHz) Motor PWM frequency
#define PWM_DEADTIME                (0.8)               ///< (us) Deadtime
#define MIN_WIND_TIME               (2.2)               ///< (us) Minimum sampling window of single-resistance current sampling
```

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。低压吊扇一般转速较低，可先用默认 16K 的调试；
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；

4.3 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；如果分压比大小，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)。

```
#define HW_RSHUNT                    (0.025)                    ///< (Ω) Current sampling resistance

 * @param (AMP4x)       Available in AMP_PGA_DUAL mode. The amplification gain is 4
 * @param (AMP8x)       Available in AMP_PGA_DUAL mode. The amplification gain is 8
 * @param (AMP16x)      Available in AMP_PGA_DUAL mode. The amplification gain is 16
 * @param (xxxx.x)      Available in AMP_NOMAL mode. The amplification gain is selected based on the external circuit
 */
#define HW_AMPGAIN                   (AMP8x)                    ///< AMP amplification gain selection

/**
 * @brief  Current sampling mode selection
 * @param (Single_Resistor)    Single-resistance current sampling mode
 * @param (Double_Resistor)    Double-resistance current sampling mode
 * @param (Three_Resistor)     Triple-resistance current sampling mode
 */
#define Shunt_Resistor_Mode          (Single_Resistor)

/* ------------------------------------------------------------------------------------------------------------------
                                        4. Bus Voltage Sampling Parameter
------------------------------------------------------------------------------------------------------------------ */
#define RV1                          (11.0)                     ///< (kΩ) Busbar voltage divider resistance 1
#define RV2                          (0.0)                      ///< (kΩ) Busbar voltage divider resistance 2
#define RV3                          (1.0)                      ///< (kΩ) Busbar voltage divider resistance 3
#define VC1                          (1.0)                      ///< Voltage compensation coefficient
#define RV                           (12.0)                     ///< voltage division ratio
```

其中，

1) 母线分压比 = (RV1 + RV2 + RV3)/RV3；

2) VC1 为电压补偿系数，只应用在启动时刻，可先不动。

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。
2. 设置过欠压保护跟保护恢复参数，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)。

```
#define HardwareCurrent_Protect        (Hardware_CMP_Protect)              ///< Hardware overcurrent protection implementation

/* -----Hardware overcurrent protection comparison value source----- */
#define Compare_DAC                    (0)                                 ///< DAC Sets the hardware overcurrent value
#define Compare_Hardware               (1)                                 ///< Hardware Sets the hardware overcurrent value
#define Compare_Mode                   (Compare_DAC)                       ///< Source of hardware overstream value

#define OverHardcurrentValue           (5.0)                               ///< (A) Hardware overcurrent value in DAC mode

#define AMP0_OffSet                    (1)                                 ///< 0: AMP0 has no bias resistance; 1: AMP0 has a bias resistanc

/* -----Protection enable----- */
#define VoltageProtectEnable           (1)                                 ///< Voltage protection, 0, disable; 1: Enable the function
#define StallProtectEnable             (1)                                 ///< Stoppage protection, 0, disabled; 1: Enable the function
#define PhaseLossProtectEnable         (1)                                 ///< Phase loss protection, 0, disable; 1: Enable the function

/* -----Over and under voltage protection----- */
#define Over_Protect_Voltage            (30)                               ///< (V) Dc voltage overvoltage protection value
#define Over_Recover_Vlotage            (28)                               ///< (V) Dc voltage overvoltage protection recovery value
#define Under_Protect_Voltage           (15)                               ///< (V) Dc voltage undervoltage protection value
#define Under_Recover_Vlotage           (17)                               ///< (V) Dc voltage undervoltage protection recovery value

/* -----lost phase protection----- */
#define PhaseLossCurrentValue          I_Value(0.2)                        ///< (A)  Lack of phase current value
#define PhaseLossRecoverTime            (200)                              ///< (1->5ms) Phase loss protection time
#define PhaseLossRestartTimes            (5)                               ///< Number of times that phase loss protection restarts (unit: t

/* -----Blocked turn protection----- */
#define MOTOR_SPEED_STAL_MAX_RPM       (400.0)                             ///< (RPM) Blocked rotating speed protection
#define MOTOR_SPEED_STAL_MIN_RPM       (10.0)
#define StallRecoverTime               (200)                               ///< (1->5ms) Stoppage protection time
#define StallProtectRestartTimes         (5)                               ///< Number of times that the lockup protection restarts (unit: t

#endif
```

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

```
#define ID_Align_CURRENT               I_Value(0.0)                        ///< (A) d-Axis desired location current
#define IQ_Align_CURRENT               I_Value(1.5)                        ///< (A) q-Axis desired location current

/* -----Startup current----- */
#define ID_Start_CURRENT               I_Value(0.0)                        ///< (A) d-Axis startup current
#define IQ_Start_CURRENT               I_Value(2.5)                        ///< (A) q-Axis startup current

/* -----Current when the controller switch to model----- */
#define ID_RUN_CURRENT                 I_Value(0.00)                       ///< (A) d-Axis running current
#define IQ_RUN_CURRENT                 I_Value(2.5)                        ///< (A) q-Axis running current
```

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，不能过小否则启动力矩太小导致启动失败。

IQ_Start_CURRENT，不能过大否则启动过冲还会引入启动噪声。

2. 切换电流: IQ_RUN_CURRENT 只决定一瞬间的电流。通过实际观测相电流，可通过 IO 口翻转确认在切环瞬间是否存在电流不平滑，可以适当调整 IQ_RUN_CURRENT 解决；
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 是(速度带宽滤波值)，以限制 FOC 估算器的最大转速输出；

```
Customer.h
124
125 /* -----Current when the controller switch to model----- */
126 #define ID_RUN_CURRENT              I_Value(0.00)          ///< (A) d-Axis running current
127 #define IQ_RUN_CURRENT              I_Value(2.5)           ///< (A) q-Axis running current
128 #define IQ_TailWind_CURRENT         I_Value(0.6)           ///< (A) q-Axis TailWind current
129
130 /* -----ATO bandwidth----- */
131 #define ATO_BW                      5.0                    ///< (Hz) ATO bandwidth, the recommended value is 1.0 - 200.0. In
132
133 #define ATO_BW_RUN                  12.0
```

注意事项:

对于低压吊扇而言，启动的第一个 ATO 影响比较明显，需要根据实际情况调整。

4. 速度带宽滤波值 SPD_BW；

```
Customer.h
133 #define ATO_BW_RUN                  12.0
134
135 /* -----Speed filter bandwidth----- */
136 #define SPD_BW                      (15.0)                 ///< (Hz) Speed filter bandwidth, the recommended value is 5.0 - 4
```

注意事项:

SPD_BW 一般不需要调整。

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
142 /* -----OMEGA startup parameter----- */
143 #define Motor_Omega_Ramp_ACC        (8.0)                  ///< Forced speed increment step in standstillstate
144 #define MOTOR_OMEGA_ACC_MIN         (20.0)                 ///< (RPM) The minimum speed to switch form forced speed to estima
145 #define MOTOR_OMEGA_ACC_END         (30.0)                 ///< (RPM) The maximum speed of forced speed
146
147 /* -----切环速度设置----- */
148 #define MOTOR_LOOP_RPM              (80.0)                 ///< (RPM) The motor speed to switch from mode0 to mode1. i.e., Th
149
150 /* -----Back EMF filter minimum bandwidth----- */
151 #define MOTOR_SPEED_SMOMIN_RPM      (50.0)                 ///< (RPM) Back EMF filter minimum bandwidth. Only available in SM
152
```

注意事项:

1) Motor_Omega_Ramp_ACC 参考值范围 1 ~ 50
2) MOTOR_OMEGA_ACC_MIN 参考值范围 1 ~ 10
3) MOTOR_OMEGA_ACC_END 参考值范围 10 ~ 80
4) MOTOR_LOOP_RPM 需要大于 MOTOR_OMEGA_ACC_END

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
160 /* -----Current loop coefficient at startup state----- */
161 #define DQKPStart                   _Q12(0.005)            ///< KP of current loop in startup state
162 #define DQKIStart                   _Q15(0.001)            ///< KI of current loop in startup state
163
164 /* -----Current loop coefficient after controller switch to model----- */
165 #define DQKP                        _Q12(0.6)              ///< KP of current loop after controller switch to model
166 #define DQKI                        _Q15(0.06)             ///< KI of current loop after controller switch to model
167
```

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 3.0 ~ 0.1
4) DQKI 建议范围 0.05 ~ 0.001

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
169 #define DOUTMAX                     _Q15(0.45)             ///< d-Axis voltage reference maximum value. The recommended value
170 #define DOUTMIN                     _Q15(-0.45)            ///< d-Axis voltage reference minimum value. The recommended value
171
172 /* -----q-Axis voltage reference limit value----- */
173 #define QOUTMAX                     _Q15(0.92)             ///< q-Axis voltage reference maximum value. The recommended value
174 #define QOUTMIN                     _Q15(-0.92)            ///< q-Axis voltage reference minimum value. The recommended value
175
```

注意事项:

1) FOC__UQ 反馈电机已经输出是否饱和
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

```
/**
 * @brief PWM Output Mode
 * @param (IPMtest)       IPM Test Mode
 * @param (NormalRun)     Nomal Output
 */
#define IPMState                        (NormalRun)
```

将IPMState选择IPMtest模式，此时UVW三相会有固定的PWM波形输出，则硬件驱动电路正常。若没有输出，则要查找硬件问题。

4.7 调试速度环

1. 将环路选择为速度环；

Customer.h*

```
 * @param (POWER_LOOP_CONTROL)      power loop
 * @param (SPEED_LOOP_CONTROL)      speed loop
 */
#define Motor_Speed_Control_Mode        (SPEED_LOOP_CONTROL)
```

2. 调速方式先选为直接给定值 NONEMODE 模式，给定档位 IRControl.TargetSpeedt 为 6 档，对应目标转速为 6 档转速 MOTOR_SPEED_LEVEL_6；

Customer.h*

```
/**
 * @brief Speed regulating mode
 * @param (NONEMODE)       Fixed speed
 * @param (OTHERMODE)      RF Control Speed
 */
#define SPEED_MODE                      (NONEMODE)
```

main.c*

```
        #if (!StartONOFF_Enable)
        {
            #if (SPEED_MODE == NONEMODE)
            {
                mcSpeedRamp.FlagONOFF = 1;
                IRControl.FlagSpeed = 1;
                IRControl.ONOFFStatus = 1;
                IRControl.TargetSpeed = 6;   ///< 档位
                mcFRState.OldTargetSpeed = IRControl.TargetSpeed;
            }
```

Customer.h

```
#define Speed_K                         (1.015)
#define MOTOR_SPEED_LEVEL_6             _Q15(260.0*Speed_K/MOTOR_SPEED_BASE)        ///< 6 gears
#define MOTOR_SPEED_LEVEL_5             _Q15(244.0*Speed_K/MOTOR_SPEED_BASE)        ///< 5 gears
#define MOTOR_SPEED_LEVEL_4             _Q15(228.0*Speed_K/MOTOR_SPEED_BASE)        ///< 4 gears
#define MOTOR_SPEED_LEVEL_3             _Q15(212.0*Speed_K/MOTOR_SPEED_BASE)        ///< 3 gears
#define MOTOR_SPEED_LEVEL_2             _Q15(196.0*Speed_K/MOTOR_SPEED_BASE)        ///< 2 gears
#define MOTOR_SPEED_LEVEL_1             _Q15(180.0*Speed_K/MOTOR_SPEED_BASE)        ///< 1 gears
#define MOTOR_SPEED_LEVEL_0             _Q15(0.0/MOTOR_SPEED_BASE)                  ///< stop
```

Customer.h

```
184
185 /* -----Set the outer loop time parameter----- */
186 #define SPEED_LOOP_TIME              (40)                    ///< (ms) Outer loop control period
187
188
189 /* -----Outer loop coefficient----- */
190 #define SKP                          _Q10(0.8)               ///< KP of Outer loop
191 #define SKI                          _Q15(0.01)              ///< KI of Outer loop
192
193 /* -----Limit value of output----- */
194 #define SOUTMAX                      I_Value(2.5)            ///< (A) Outer loop output maximum value
195 #define SOUTMIN                      I_Value(0.02)           ///< (A) Outer loop output minimum value
196
197 /* -----Speed Loop increment and decrease setting----- */
198 #define SPEED_INC                    (1.0)                   ///< Increment step value in speed control mode
199 #define SPEED_DEC                    (1.0)                   ///< Decrement step value in speed control mode
200
201 /* -----Power Loop increment and decrease setting----- */
202 #define POWER_INC                    (1.0)                   ///< Increment step value in power control mode
203 #define POWER_DEC                    (1.0)                   ///< Decrement step value in power control mode
204
```

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。
   - 启动到切环路的电流: IQ_RUN_CURRENT，给到稍微比启动电流小点即可
   - 影响启动频率的 ATO 和 Omega 的参数等等
4. 当上电，电机能跑后，调整 PI 外环最大限幅值 SOUTMAX，外环 KP、KI，让实际转速稳定在目标转速上；

常见问题即解决办法:

1) 调整 PI 外环最大限幅值 SOUTMAX，还是达不到目标转速；
   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 调试功率环

1. 环路选择为功率环；

Customer.h

```
216  * @brief Mode selection sets the value
217  * @param (POWER_LOOP_CONTROL)      power loop
218  * @param (SPEED_LOOP_CONTROL)      speed loop
219  */
220 #define Motor_Speed_Control_Mode       (POWER_LOOP_CONTROL)
221
```

2. 调速方式先选为直接给定值 NONEMODE 模式，给定档位 IRControl.TargetSpeedt 为 6

档，对应目标转速为 6 档转速 MOTOR_POWER_LEVEL_6；

Customer.h*

```
/**
 * @brief Speed regulating mode
 * @param (NONEMODE)       Fixed speed
 * @param (OTHERMODE)      RF Control Speed
 */
#define SPEED_MODE                      (NONEMODE)
```

main.c*

```
        #if (!StartONOFF_Enable)
        {
            #if (SPEED_MODE == NONEMODE)
            {
                mcSpeedRamp.FlagONOFF = 1;
                IRControl.FlagSpeed = 1;
                IRControl.ONOFFStatus = 1;
                IRControl.TargetSpeed = 6;   ///< 档位
                mcFRState.OldTargetSpeed = IRControl.TargetSpeed;
            }
```

Customer.h

```
#define MOTOR_POWER_LEVEL_6          _Q15((30.0)/MOTOR_POWER_BASE*POWER_REG_K)   ///< 6 gears
#define MOTOR_POWER_LEVEL_5          _Q15((26.0)/MOTOR_POWER_BASE*POWER_REG_K)   ///< 5 gears
#define MOTOR_POWER_LEVEL_4          _Q15((22.0)/MOTOR_POWER_BASE*POWER_REG_K)   ///< 4 gears
#define MOTOR_POWER_LEVEL_3          _Q15((18.0)/MOTOR_POWER_BASE*POWER_REG_K)   ///< 3 gears
#define MOTOR_POWER_LEVEL_2          _Q15((15.0)/MOTOR_POWER_BASE*POWER_REG_K)   ///< 2 gears
#define MOTOR_POWER_LEVEL_1          _Q15((11.0)/MOTOR_POWER_BASE*POWER_REG_K)   ///< 1 gears
#define MOTOR_POWER_LEVEL_0          _Q15(0/MOTOR_POWER_BASE*POWER_REG_K)        ///< stop
```

3. 通过调整功率环 PI(SKP 和 SKI)和功率环爬坡增量保证功率环稳定，启动响应快且不过冲。

Customer.h

```
/* -----Set the outer loop time parameter----- */
#define SPEED_LOOP_TIME              (40)                    ///< (ms) Outer loop control period

/* -----Outer loop coefficient----- */
#define SKP                          _Q10(0.8)               ///< KP of Outer loop
#define SKI                          _Q15(0.01)              ///< KI of Outer loop

/* -----Limit value of output----- */
#define SOUTMAX                      I_Value(2.5)            ///< (A) Outer loop output maximum value
#define SOUTMIN                      I_Value(0.02)           ///< (A) Outer loop output minimum value

/* -----Speed Loop increment and decrease setting----- */
#define SPEED_INC                    (1.0)                   ///< Increment step value in speed control mode
#define SPEED_DEC                    (1.0)                   ///< Decrement step value in speed control mode

/* -----Power Loop increment and decrease setting----- */
#define POWER_INC                    (1.0)                   ///< Increment step value in power control mode
#define POWER_DEC                    (1.0)                   ///< Decrement step value in power control mode
```

说明:

一般来说此时计算得到的 mcFocCtrl.Powerlpf 跟真实功率是成正比关系，所以我们需要先测量一组 mcFocCtrl.Powerlpf 和真实功率 Power，计算得到功率系数 POWER_REG_K。

例如当电机跑 100W 的时候，mcFocCtrl.Powerlpf 此时为 5000，那么功率系数 POWER_REG_K = mcFocCtrl.Powerlpf/真实功率值 = 50；那么如果设置目标值为 500*50，即相当于目标值为 500W。

功率环常见问题:

1) 相电流抖动。

解决:主要调整功率环的 PI，或者功率环反馈值 mcFocCtrl.Powerlpf 的滤波系数。

AddFunction.c

```
    static int16 TempEsValue = 0;

    /* -----功率滤波----- */
    LPF_MDU((FOC__POW << 1), 10, mcFocCtrl.Powerlpf, mcFocCtrl.Powerlpf_LSB); ///< 注意低通滤波器系数范围为0---127
```

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。

4.8.1 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。

4.8.2 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. 将 StartONOFF_Enable 打开；
2. 根据实际情况配置运行时间 StartON_Time 和停止时间 StartOFF_Time；
3. 调整 mcSpeedRamp.TargetValue 的值可以修改启停的功率大小；
4. 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；
5. 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)。

Customer.h

```
271                                  ON-OFF Test parameter
272  ------------------------------------------------------------------------------------------------------------------
273  #define StartONOFF_Enable          (0)
274  #define StartON_Time               (2000)                    ///< (ms) Start up run time
275  #define StartOFF_Time              (18000)                   ///< (ms) Stop Time
276
```

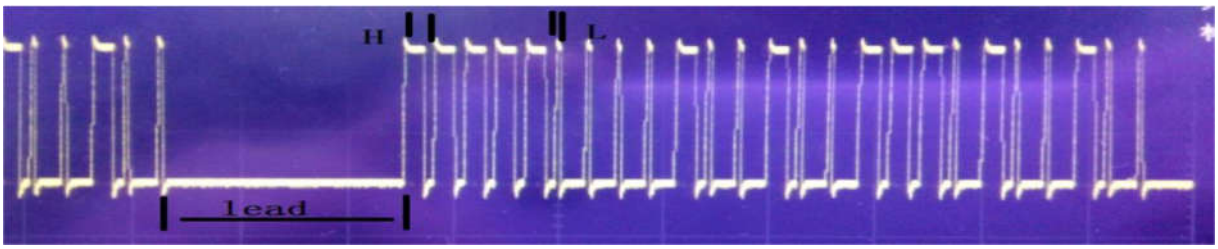
4.8.3 调速方式选择射频调速

1. 当启停都验证没问题了，调速方式选择射频调速模式。

Customer.h

```
223   * @brief Speed regulating mode
224   * @param (NONEMODE)       Fixed speed
225   * @param (OTHERMODE)      RF Control Speed
226   */
227  #define SPEED_MODE                    (OTHERMODE)
```

2. 测试射频信号，完成解码。



IRScan.h*

```
13
14  #define   ShortPressMode              (0)
15  #define   LongPressMode               (1)
16  #define   UserCodeStudyMode           (LongPressMode)              ///< 遥控学习模式：ShortPressMode:短按某个键执行学习；
17                                                                     ///<LongPressMode：长按某个键学习
18
19
20  #define   UserCodeLength          (20)            ///<定义遥控地址码长度
21  #define   DataCodeLength          (5)             ///<定义遥控数据码长度
22  #define   CheckCodeLengh          (6)
23  #define   AllCodeLength           (UserCodeLength + DataCodeLength + CheckCodeLengh)       ///<定义遥控数据码长度
24
25  /*IR Scan Parameter*/
26  #define   IRBitValue1TimeMax            (1.0)           ///< 数据为1时高电平长度最大值，单位ms
27  #define   IRBitValue1TimeMin            (0.6)           ///< 数据为1时高电平长度最小值，单位ms
28  #define   IRBitValue0TimeMax            (0.45)          ///< 数据为0时高电平长度最大值，单位ms
29  #define   IRBitValue0TimeMin            (0.15)          ///< 数据为0时高电平长度最小值，单位ms
30  #define   IRLeadCodeTimeMax             (10.0)          ///< 引导码周期长度最大值，单位ms
31  #define   IRLeadCodeTimeMin             (6.0)
```

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

选择 Omega 启动方式。。

Customer.h

```
259  #define FRDetectMethod                (BEMFFOCMethod)                  ///<
260
261  /**
262    * @brief Start Mode
263    * @param (Open_Start)       Open loop strong drag start
264    * @param (Omega_Start)      Omega start
265    * @param (Open_Omega_Start) Open loop strong first, and thaen Omega start
266    */
267  #define Open_Start_Mode               (Omega_Start)
268
```

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

```
142  /* -----OMEGA startup parameter----- */
143  #define Motor_Omega_Ramp_ACC          (8.0)                ///< Forced speed increment step in standstillstate
144  #define MOTOR_OMEGA_ACC_MIN           (20.0)               ///< (RPM) The minimum speed to switch form forced speed to estima
145  #define MOTOR_OMEGA_ACC_END           (30.0)               ///< (RPM) The maximum speed of forced speed
146
147  /* -----切环速度设置----- */
148  #define MOTOR_LOOP_RPM                (80.0)               ///< (RPM) The motor speed to switch from mode0 to mode1. i.e., Th
149
150  /* -----Back EMF filter minimum bandwidth----- */
151  #define MOTOR_SPEED_SMOMIN_RPM        (50.0)               ///< (RPM) Back EMF filter minimum bandwidth. Only available in SM
152
```

启动过程如下图所示:

5.1.2 PLL_Start 启动

```
307  * @brief 启动模式选择
308  * @param (Open_Start)        开环强拖启动
309  * @param (Omega_Start)       Omega启动
310  * @param (Open_Omega_Start) 先开环强拖，后Omega启动
311  * @param (PLL_Start)         Software PLL start
312  */
313 #define Open_Start_Mode                (PLL_Start)
```

选择 PLL_Start 启动方式。

```
172 /*-------软件PLL_Start启动参数-----------------------------------------------------------------*/
173 #define BemfKeEnable                (0)                  ///< 反电动势Ke识别使能，0不使能，1使能
174 #define ClimbingFactorMin           (0.5)                ///<  启动PLL算法里爬坡系数最小值,经典值 0.1~8.0
175 #define ClimbingFactorMax           (4.0)                ///<  启动PLL算法里爬坡系数最大值,经典值 0.1~8.0
176 #define IncClimbingFactor           (10)                 ///<  启动PLL算法里爬坡增量,经典值 1~15
177 #define Focred_Time                 (11000)              ///<  启动PLL算法里识别时间，单位ms <40000
178 #define PLL_Pwm_Switch              (15000)              ///< 启动软件PLL运行载波中断个数
179 #define PLL_Speed_Switch            (48.8)               ///< (RPM)软件PLL切内部估算器速度
180 #define Speed_PLL_EOME_Deta         (0)                  ///< (RPM)软件PLL切内部估算器速度,EOME速度误差
```

静止启动时，用 PLL 观测器启动，当电机转速大于或等于 PLL_Speed_Switch 时，切内部硬件 SMO 观测器运行。影响启动的关键参数是 ClimbingFactorMin、ClimbingFactorMax 和 IncClimbingFactor。电机启动慢，三个参数可以相应改大，反之则改小。如需要电机堵住松手不失步效果，需要注意参数 ClimbingFactorMax 不能太大，针对效果调整。参数 IncClimbingFactor 是参数 ClimbingFactorMin 递增到 ClimbingFactorMax 的增量，影响启动快慢。

5.1.3 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，此时依次加大 ATO_BW、ATO_BW_RUN； 3.检查硬件电路运放 AMPO 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。 5.启动 PLL 算法的系数 ClimbingFactor 太小，适当改大。
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； 2. Omega 启动参数也会有影响。 3. 启动 PLL 算法的系数 ClimbingFactor 太大，适当改小。
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_RUN，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。 3. 启动 PLL 算法的系数 ClimbingFactor 太小，适当改大。

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。 3. 启动 PLL 算法的系数 ClimbingFactor 太大，适当改小。

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陷保护触发不了，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。

Protect.H

```
#define Compare_DAC                 (0)                 ///< DAC Sets the hardware overcurrent value
#define Compare_Hardware            (1)                 ///< Hardware Sets the hardware overcurrent value
#define Compare_Mode                (Compare_DAC)       ///< Source of hardware overstream value

#define OverHardcurrentValue        (5.0)               ///< (A) Hardware overcurrent value in DAC mode
```

5.2.2 电压保护

程序通过AD14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。

Protect.H

```
#define StallProtectEnable          (1)                 ///< Stoppage protection, 0, disabled; 1: Enable the function
#define PhaseLossProtectEnable      (1)                 ///< Phase loss protection, 0, disable; 1: Enable the function

/* -----Over and under voltage protection----- */
#define Over_Protect_Voltage        (30)                ///< (V) Dc voltage overvoltage protection value
#define Over_Recover_Vlotage        (28)                ///< (V) Dc voltage overvoltage protection recovery value
#define Under_Protect_Voltage       (15)                ///< (V) Dc voltage undervoltage protection value
#define Under_Recover_Vlotage       (17)                ///< (V) Dc voltage undervoltage protection recovery value
```

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2倍，且该相最大电流大于设定的PhaseLossCurrentValue值，则判定为缺相。

Protect.H

```
#define StallProtectEnable              (1)                     ///< Stoppage protection, 0, disabled; 1: Enable the function
#define PhaseLossProtectEnable          (1)                     ///< Phase loss protection, 0, disable; 1: Enable the functio

/* -----Over and under voltage protection----- */
#define Over_Protect_Voltage             (30)                   ///< (V) Dc voltage overvoltage protection value
#define Over_Recover_Vlotage             (28)                   ///< (V) Dc voltage overvoltage protection recovery value
#define Under_Protect_Voltage            (15)                   ///< (V) Dc voltage undervoltage protection value
#define Under_Recover_Vlotage            (17)                   ///< (V) Dc voltage undervoltage protection recovery value

/* -----lost phase protection----- */
#define PhaseLossCurrentValue           I_Value(0.2)            ///< (A)  Lack of phase current value
#define PhaseLossRecoverTime            (200)                   ///< (1->5ms) Phase loss protection time
#define PhaseLossRestartTimes            (5)                    ///< Number of times that phase loss protection restarts (uni
```

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

堵转保护有两种方法检测：

1. 判断反电动势太小或当反电动势太小，转速太大-

MotorProtect.c

```
    {
        /* -----method 1: 判断反电动势太小或当反电动势太小，转速太大----- */
        if ((mcFocCtrl.EsValue < 20) || (mcFocCtrl.EsValue > 0xFFF0) || ((FOC__EOME > _Q15(100.0 / MOTOR_SPEED_BASE)) && (mcFocCtrl.EsValu
        {
            mcFaultDect.StallDectEs++;

            if (mcFaultDect.StallDectEs >= 200)
            {
                mcFaultDect.StallDectEs = 0;
                mcFaultSource           = FaultStall;
                mcProtectTime.StallTimes++;
                mcProtectTime.StallFlag = 1;
            }
        }
        else
        {
            if ( mcFaultDect.StallDectEs > 0)
            {
                mcFaultDect.StallDectEs--;
            }
        }
```

2. 通过检测估算转速，当估算转速超过设定转速MOTOR_SPEED_STAL_MAX_RPM，或者低于设定转速MOTOR_SPEED_STAL_MIN_RPM，则认为发生了堵转；

MotorProtect.c

```
230
231             /* -----method 2, 判断速度低于堵转最小值或者超过堵转最大值----- */
232             if ((mcFocCtrl.SpeedFlt <  _Q15(50.0 / MOTOR_SPEED_BASE)) || (mcFocCtrl.SpeedFlt > Motor_Stall_Max_Speed))
233             {
234                 mcFaultDect.StallDectSpeed++;
235
236                 if (mcFaultDect.StallDectSpeed >= 5)
237                 {
238                     mcFaultDect.StallDectSpeed = 0;
239                     mcFaultSource           = FaultStall;
240                     mcProtectTime.StallTimes++;
241                     mcProtectTime.StallFlag = 2;
242                 }
243             }
244             else
245             {
246                 if (mcFaultDect.StallDectSpeed > 0)
247                 {
248                     mcFaultDect.StallDectSpeed--;
249                 }
250             }
```

5.2.5 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

Customer.h

```
298 #define MOTOR_POWER_LEVEL_3             _Q15((18.0)/MOTOR_POWER_BASE*POWER_REG_K)    ///< 3 gears
299 #define MOTOR_POWER_LEVEL_2             _Q15((15.0)/MOTOR_POWER_BASE*POWER_REG_K)    ///< 2 gears
300 #define MOTOR_POWER_LEVEL_1             _Q15((11.0)/MOTOR_POWER_BASE*POWER_REG_K)    ///< 1 gears
301 #define MOTOR_POWER_LEVEL_0             _Q15(0/MOTOR_POWER_BASE*POWER_REG_K)         ///< stop
302
303 /* -----Define the rotational speed for forward and reverse switching----- */
304 #define MOTOR_SPEED_LEVEL_7             _Q15(130.0/MOTOR_SPEED_BASE)                  ///< (RPM)The speed ring detects the brake speed at rever
305 #define MOTOR_POWER_LEVEL_7             _Q15((0.0)/MOTOR_POWER_BASE*POWER_REG_K)      ///< (RPM)The power ring detects the target power at inve
306
307 /* -----Reverse and reverse switch brake time Settings----- */
308 #define MOTOR_BrakeTime                 (7000)                                        ///< (ms)The detection is the braking time in reverse
309
310 /* -----Power limit setting----- */
311 #define POWER_LIMIT_ENALBE              (1)                                           ///<   Power limit setting,1 Enable, 0 Disable
312 #define MOTOR_LIMIT_POWER               (30.5)                                        ///<(W) Limit the power target value
313 #define POWER_LIMIT_VALUE               _Q15(MOTOR_LIMIT_POWER/MOTOR_POWER_BASE*POWER_REG_K)
314 #define MOTOR_POWER_BASE                (80.0)                                        ///<(W) Maximum limit power
315 #define POWER_REG_K                     (0.22)                                        ///< Ratio coefficient between FOC POW register value and a
316
```

6.2 限速功能

Customer.h*

```
298 #define MOTOR_POWER_LEVEL_3             _Q15((18.0)/MOTOR_POWER_BASE*POWER_REG_K)    ///< 3 gears
299 #define MOTOR_POWER_LEVEL_2             _Q15((15.0)/MOTOR_POWER_BASE*POWER_REG_K)    ///< 2 gears
300 #define MOTOR_POWER_LEVEL_1             _Q15((11.0)/MOTOR_POWER_BASE*POWER_REG_K)    ///< 1 gears
301 #define MOTOR_POWER_LEVEL_0             _Q15(0/MOTOR_POWER_BASE*POWER_REG_K)         ///< stop
302
303 /* -----Define the rotational speed for forward and reverse switching----- */
304 #define MOTOR_SPEED_LEVEL_7             _Q15(130.0/MOTOR_SPEED_BASE)                  ///< (RPM)The speed ring detects the brake speed at reve
305 #define MOTOR_POWER_LEVEL_7             _Q15((0.0)/MOTOR_POWER_BASE*POWER_REG_K)      ///< (RPM)The power ring detects the target power at inv
306
307 /* -----Reverse and reverse switch brake time Settings----- */
308 #define MOTOR_BrakeTime                 (7000)                                        ///< (ms)The detection is the braking time in reverse
309
310 /* -----Power limit setting----- */
311 #define POWER_LIMIT_ENALBE              (1)                                           ///<   Power limit setting,1 Enable, 0 Disable
312 #define MOTOR_LIMIT_POWER               (30.5)                                        ///<(W) Limit the power target value
313 #define POWER_LIMIT_VALUE               _Q15(MOTOR_LIMIT_POWER/MOTOR_POWER_BASE*POWER_REG_K)
314 #define MOTOR_POWER_BASE                (80.0)                                        ///<(W) Maximum limit power
315 #define POWER_REG_K                     (0.22)                                        ///< Ratio coefficient between FOC POW register value and
316
317 /* -----Speed limit setting----- */
318 #define SPEED_LIMIT_ENALBE              (1)                                           ///< Speed limit setting,1 Enable, 0 Disable
319 #define MOTOR_SPEED_LIMIT_RPM           (280.0)                                       ///< Limit speed (RPM)
320
```

7 修改记录

版本号：第1位-原理 第2位-模块 第3和4位-细节

版本号	修改详细内容说明	生效日期	修订者	审核者
V1.0.00	初稿	2022-07-07	Harvey.Song	
V1.0.01	调整图片以及启动参数	2022-11-22	Harvey.Song	
V1.0.02	框图添加 mcBemfCheck	2022-12-09	Harvey.Song	

8 版权说明

版权所有©峰岹科技（深圳）股份有限公司（以下简称：峰岹科技）。

为改进设计和/或性能，峰岹科技保留对本文档所描述或包含的产品（包括电路、标准元件和/或软件）进行更改的权利。本文档中包含的信息供峰岹科技的客户进行一般性使用。峰岹科技的客户应确保采取适当行动，以使其对峰岹科技产品的使用不侵犯任何专利。峰岹科技尊重第三方的有效专利权，不侵犯或协助他人侵犯该等权利。
本文档版权归峰岹科技所有，未经峰岹科技明确书面许可，任何单位及个人不得以任何形式或方式（如电子、机械、磁性、光学、化学、手工操作或其他任何方式），对本文档任何内容进行复制、传播、抄录、存储于检索系统或翻译为任何语种，亦不得更改或删除本内容副本中的任何版权或其他声明信息。

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