参考文献

Design Guide: TIDA-010265

采用 C2000™ 和 MSPM0 的 750W 电机逆变器参考设计

说明

该参考设计是一款适用于洗衣机或类似应用的 750W 电机驱动器，展示了一种通过 FAST™ 软件编码器或 eSMO 为三相 PMSM 实现无传感器 FOC 控制的方法。该参考设计采用模块化设计，支持 C2000™ MCU 和 MSPM0 系列微控制器子板位于同一主板上。此参考设计提供的硬件和软件已经过测试，而且可随时使用，有助于加快开发，从而缩短产品上市时间。该设计指南提供了硬件设计详细信息和测试结果。

资源

TIDA-010265	设计文件夹
TMS320F2800137	产品文件夹
MSPM0G1507	产品文件夹
UCC28881、TPS54202、TLV9062	产品文件夹
C2000WARE-MOTORCONTROL-SDK	工具文件夹

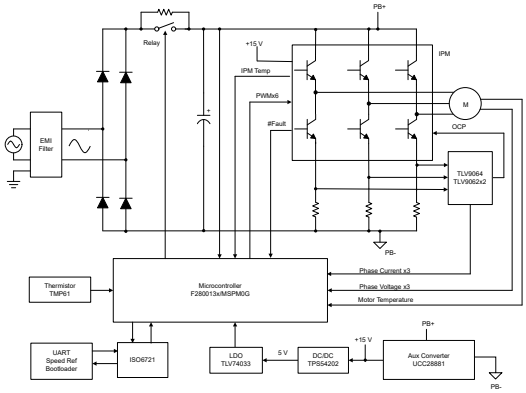
请咨询我司 TI E2E™ 支持专家

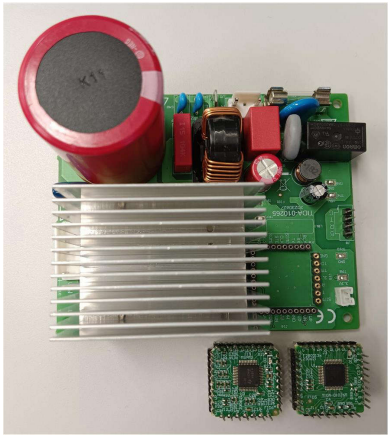
特性

- 宽工作电压输入范围：165V 至 265V 交流，50/60Hz。
- 具有高达 750W 的逆变器级、15kHz 开关频率、扭矩补偿和自动弱磁控制
- C2000 或 MSPM0 控制器子板位于同一电源主板上的模块化设计
- 无传感器磁场定向控制 (FOC) 电机控制，支持 FAST 和 eSMO 观测器
- 用户友好型图形用户界面 (GUI)，用于控制、识别和监控电机

应用

- 洗衣机和烘干机
- 空调室内机
- 冰箱和冷冻柜
- 电器：压缩机





1 系统说明

当今大型家电或类似应用所采用的电机控制技术必须满足日益增长的对于更低成本、更小尺寸、更高功率和更高能效的需求。磁同步电机 (PMSM) 在大型家电应用中越来越常用。

该参考设计提供单个 750W 逆变器主板，并通过 TMS320F2800137 和 MSPM0G1507 子卡进行控制，因此用户可以方便地使用该参考设计来在同一硬件平台上评估 C2000 和 MSPM0 系列微控制器。

软件支持 FAST 和 eSMO 观测器，因此可以比较这两个器件的性能。用户友好型 GUI 还有助于识别电机并调整控制参数，从而加快开发速度。

1.1 术语

SLYZ022	TI 术语表：本术语表列出并解释了术语、首字母缩略词和定义
PMSM	永久磁性同步电机
BLDC	无刷直流
BEMF	反电动势
PWM	脉宽调制
FET、MOSFET	金属氧化物半导体场效应晶体管
IGBT	绝缘栅双极晶体管
RMS	均方根
MTPA	每安培最大扭矩
FWC	弱磁控制
FOC	场定向控制
HVAC	暖通空调
ESMO	增强型滑模观测器
PLL	锁相环
FAST	磁通、角度、转速和扭矩观测器

1.2 关键系统规格

表 1-1 中列出了 TIDA-010265 规格。

表 1-1. 关键系统规格

参数	测试条件	最小值	标称值	最大值	单位
	系统输入特性				
输入电压 (V_{INAC})	-	165	230	265	VAC
输入频率 (f_{LINE})	-	47 Ω	50	63	Hz
空载待机功耗 (P_{NL})	V_{INAC} = 230V，I_{out} = 0A	-	3.0	-	W
输入电流 (I_{IN})	V_{INAC} = 230V，I_{out} = I_{MAX}	-	8	-	A
	电机逆变器特性				
PWM 开关频率 (f_{SW})	-	-	15	20	kHz
额定输出功率 (P_{OUT})	V_{INAC} = 标称值	-	500	750	W
输出电流 (I_{RMS})	V_{INAC} = 标称值	-	3	-	A
逆变器效率 (η)	V_{INAC} = 标称值，P_{OUT} = 标称值	-	98	-	%
电机电频率 (f)	V_{INAC} = 最小值至最大值	20	200	400	Hz
故障保护	过流、恢复失速、欠压、过压				
驱动控制方法和功能	采用三个或单个用于电流检测的分流电阻器的无传感器 FOC				
	系统特性				
内置辅助电源	V_{INAC} = 最小值至最大值	15V ±10%、200mA，3.3V ±10%、300mA			
工作环境	开放式框架	- 10	25	55	°C
电路板尺寸	长 × 宽 × 高	105mm × 85mm × 60mm			mm^3

警告

TI 建议该参考设计仅可在实验室环境中运行，不应将该器件作为成品供一般消费者使用。

TI 建议，该参考设计仅可由熟悉处理高压电子和机械部件、系统及子系统所存在相关风险的合格工程师和技术人员使用。

高电压！ 电路板中存在可接触的高电压。如电路板的电压和电流处理不当或施加不正确，则可能导致电击、火灾或伤害事故。使用该设备时应特别小心，并采取相应的保护措施，以避免伤害自己或损坏财产。

表面高温！接触会导致烫伤。**请勿触摸！**电路板上电后，某些元件可能会达到 55°C 以上的高温。由于存在高温，在运行过程中或运行刚结束时，用户不得触摸电路板。

小心

请勿在无人照看的情况下使该设计通电。

2 系统概述

2.1 方框图

图 2-1 所示为该参考设计的方框图。

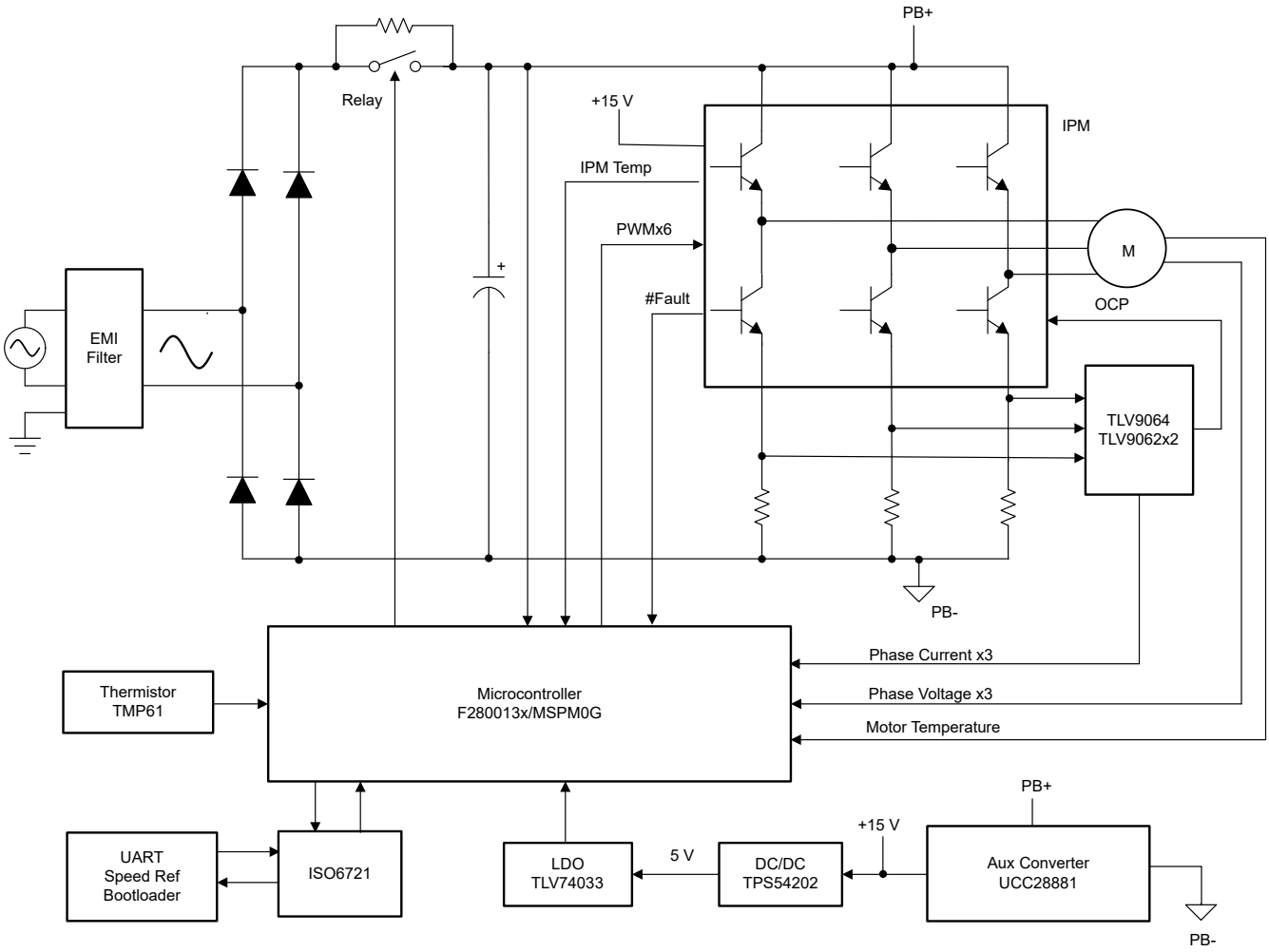


图 2-1. TIDA-010265 750W 电机逆变器方框图

整个系统可分为七个块：

- EMI 滤波器
- 桥式整流器
- 三相逆变器
- 辅助电源
- TMS320F2800137 子卡
- MSPM0G1507 子卡

2.2 设计注意事项

该设计支持使用 C2000 或 MSPM0 控制器进行单电机控制。具有高抗噪性能的电流和电压检测设计对于精确的电机驱动而言是必不可少的。以下部分详细介绍了该设计中使用的检测和驱动电路。硬件设计文件位于 C2000Ware Motor Control SDK 安装目录 `<install_location>\solutions \tida_010265_wminv\hardware` 下。

2.3 重点产品

本参考设计采用了以下重点产品。以下各节介绍为该参考设计选择器件时应考虑的主要特性。如需了解有关重点器件的更多详细信息，请参阅各自的产品数据表。

2.3.1 TMS320F2800137

TMS320F280013x 是 C2000™ 可扩展、超低延迟实时微控制器器件系列中的一款器件，专为提高电力电子应用的效率而设计。实时控制子系统基于 TI 的 32 位 C28x 数字信号处理器 (DSP) 内核，可针对从片上闪存或 SRAM 运行的浮点或定点代码提供 120MHz 的信号处理性能。三角函数加速器 (TMU) 和循环冗余校验 (VCRC) 扩展指令集进一步增强了 C28x CPU 的性能，从而加快了实时控制系统关键常用算法的速度。高性能模拟块集成在 F280013x 实时微控制器 (MCU) 上，并与处理单元和 PWM 单元紧密耦合，从而提供出色的实时信号链性能。14 个 PWM 通道均支持与频率无关的分辨率模式，可控制从三相逆变器到高级多级电源拓扑的各种功率级。连接可以通过各种业界通用通信端口（如 SPI、三个 SCI|URAT、I2C 和 CAN）进行，另外还提供了多个引脚多路复用选项，可实现出色的信号布局。

2.3.2 MSPM0G1507

MSPM0G150x 微控制器 (MCU) 属于高度集成的超低功耗 32 位混合信号处理 (MSP) MCU 系列，该系列基于增强型 Arm® Cortex®-M0+ 32 位内核平台，工作频率高达 80MHz。这些成本优化型 MCU 提供高性能模拟外设集成，支持 –40°C 至 125°C 的工作温度范围，并在 1.62V 至 3.6V 的电源电压下运行。MSPM0G150x 器件提供具有内置纠错码 (ECC) 且高达 128KB 的嵌入式闪存程序存储器以及具有硬件奇偶校验选项且高达 32KB 的 SRAM。此类器件还包含一个存储器保护单元、7 通道 DMA、数学加速器和各种高性能模拟外设，例如两个 12 位 4MSPS ADC、一个可配置内部共享电压基准、一个 12 位 1MSPS DAC、三个具有内置基准数模转换器 (DAC) 的高速比较器、两个具有可编程增益的零漂移零交叉运算放大器和一个通用放大器。此外，还提供智能数字外设，例如三个 16 位高级控制计时器、三个 16 位通用计时器、一个 24 位高分辨率计时器、两个窗口式看门狗计时器以及一个具有警报和日历模式的 RTC。这类器件可提供数据完整性和加密外设以及增强型通信接口（四个 UART、两个 I2C、两个串行外设接口 (SPI)）。

2.3.3 TMP6131

TMP61x 线性热敏电阻可在整个温度范围内提供线性度和始终如一的灵敏度，支持使用简单而准确的方法进行温度转换。该器件的低功耗和较小的热质量可充分减小自发热的影响。

这些器件具有内置的高温失效防护性能以及对环境变化的强大抵抗力，设计用于长寿命的高性能应用。TMP6 系列器件外型小巧，可靠近热源放置，并具有快速响应时间。

2.3.4 UCC28881

UCC28881 在单片器件中集成了控制器和 14Ω、700V 功率 MOSFET。该器件还集成了高压电流源，能够在经整流的市电电压下直接启动和运行。UCC28881 与 UCC28880 属于同一系列的器件，但电流更高。

该器件的静态电流较低，能够提供出色的效率。凭借 UCC28881，使用最少的外部元件即可构建降压、降压/升压以及反激拓扑等最常用的转换器拓扑。

2.3.5 TPS54202

TPS54202 是一款输入电压范围为 4.5V 至 28V 的 2A 同步降压转换器。该器件包含两个集成式开关场效应晶体管 (FET) 并且具备内部回路补偿和 5ms 内部软启动功能，可降低组件数。

高级 Eco-mode 实现可尽可能提高轻负载效率并降低功率损耗。

两个高侧 MOSFET 内的逐周期电流限制可在过载情况下保护转换器，并通过低侧 MOSFET 续流电流限制防止电流失控，增强限制效果。

2.3.6 TLV9062

TLV9062 是具有轨至轨输入和输出摆幅功能的双路低压（1.8V 至 5.5V）运算放大器。该器件是非常具有成本效益的设计，适用于需要低电压运行、小型封装尺寸和高容性负载驱动能力的应用。虽然 TLV906x 的容性负载驱动能力为 100pF，但电阻式开环输出阻抗便于在更高的容性负载下更轻松地实现稳定。TLV906xS 器件具有关断模式，允许放大器切换至典型电流消耗低于 1μA 的待机模式。TLV906xS 系列有助于简化系统设计，因为该系列具有稳定的单位增益，集成了 RFI 和 EMI 抑制滤波器，而且在过驱条件下不会出现反相。

2.3.7 TLV74033

TLV740P 低压降 (LDO) 线性稳压器是一款低静态电流 LDO，具有出色的线路和负载瞬态性能，专为对功耗敏感的应用设计。此器件可提供 1% 的典型精度。

TLV740P 还可在器件上电和使能期间提供浪涌电流控制。TLV740P 将输入电流限制为定义的电流限值，从而防止从输入电源流出的电流过大。此功能对于电池供电类器件尤为重要。

2.4 系统设计原理

该参考设计的主要重点是适用于洗衣机或类似电器应用的单电机控制。

2.4.1 硬件设计

典型的电机控制板包含多个块：辅助电源、逆变器、电流和电压检测、保护电路以及微控制器。本节将介绍这些设计概念。

2.4.1.1 模块化设计

该参考设计采用模块化设计，具有两个子板和一个主板。一个子板具有 MSPM0G1507 微控制器以及基于两个内部高端运算放大器的两相电流放大电路。另一个子板具有 TMS320F2800137 微控制器和两个 TLV9062 器件作为相电流放大器。主板上装有所有功率器件，包括交流滤波器、整流器和智能电源模块 (IPM)。图 2-2 和图 2-3 展示了主板和子板。

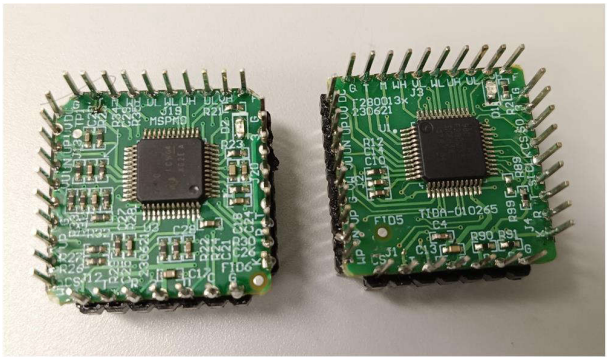


图 2-2. TIDA-010265 子板

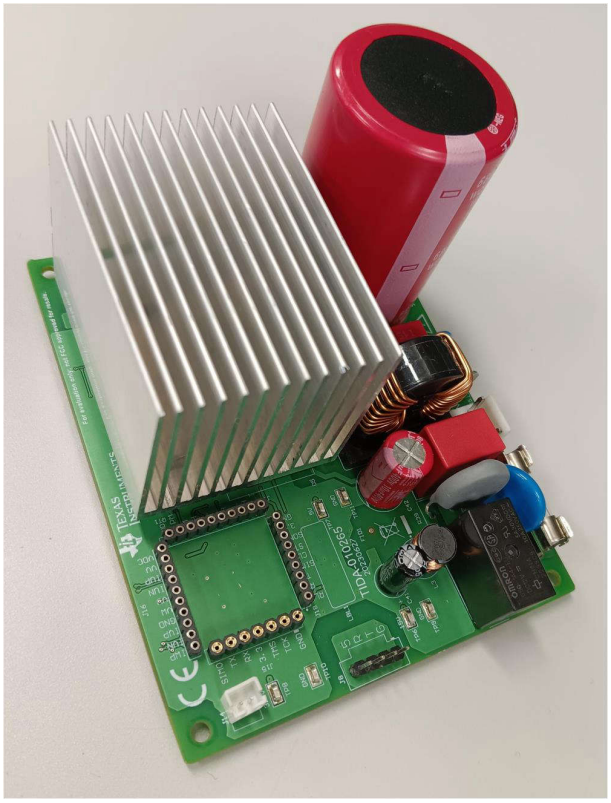


图 2-3. TIDA-010265 主板

2.4.1.2 高压降压辅助电源

基于 UCC28881 的非隔离式高压降压电源为该参考设计提供辅助电源，可在 15VDC 下提供高达 200mA 的电流。图 2-4 展示了 UCC28881 高压降压电源电路。

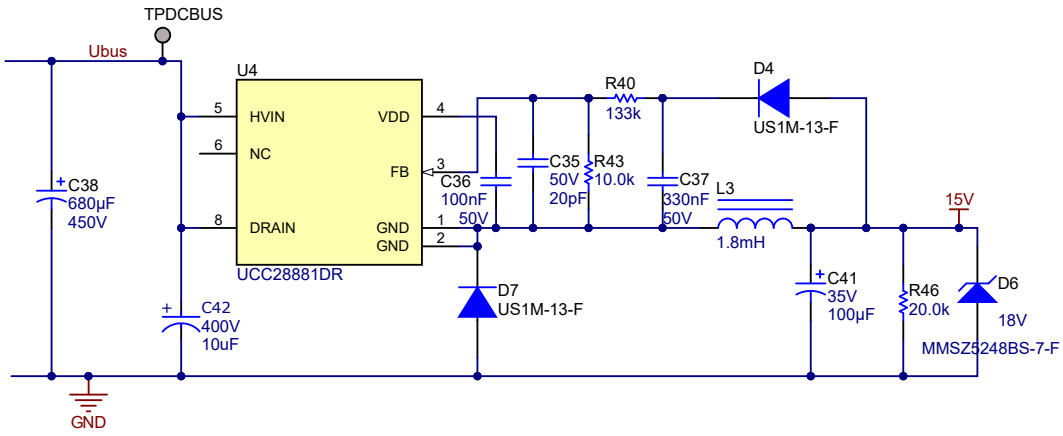


图 2-4. 高压降压电源电路

2.4.1.3 直流链路电压检测

直流电压检测电路用于将整流后的电压信号转换为可由低成本电阻网络实现的低压信号，如图 2-5 所示。直流总线电压也可用于估算交流输入电压。

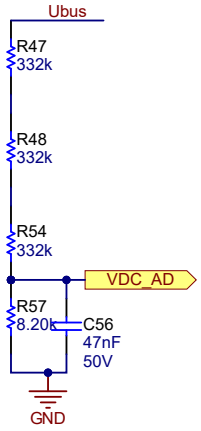


图 2-5. 直流母线电压检测电路

2.4.1.4 电机相电压检测

适用于 TIDA-010265 的 C2000 软件支持增强型滑模观测器 (eSMO) 以及磁通、角度、速度和扭矩 (FAST™) 观测器。FAST 观测器可以提高低速性能并减小速度容差；但是，除了三相电流检测之外，FAST 还需要 3 个电机相电压检测。节 2.4.2.4.2 详细介绍了电机相电压检测设计。

2.4.1.5 电机相电流检测

MS320F2800137 子板设计为支持 1-3 相电流检测，而 MSPM0G1507 子板支持 1-2 相电流检测。节 2.4.2.4.1 中详细介绍了电机相电流检测设计。

2.4.1.6 外部过流保护

该参考设计通过外部比较器和内部比较器实现过流保护 (OCP)。图 2-6 展示了具有外部比较器的 OCP，该电路汇总电流的三个相位，然后与 U10 负输入端的基准电压进行比较，以生成到 IPM 的高电压，然后由 IPM 生成过流故障信号并报告给微控制器。确切的过流保护电流可以通过以下公式计算。

$$V_{-} = \frac{3.3\ V}{R_{20} + R_{108}} \times R_{108} = \frac{3.3\ V}{20\ k + 1\ k} \times 1k = 0.1571\ V \tag{1}$$

$$V_{+} = \frac{I_{OCP} \times R_{80}}{R87 + (R_{104} + R_{105})/2)} \times \left(\left(R_{104} + R_{105}\right)/2\right) = \frac{0.05 \times I_{OCP}}{300 + 150} \times 150 = \frac{0.05 \times I_{OCP}}{3} \tag{2}$$

$$I_{ocp} = \frac{V_{+}}{R_{87}} \times 3 = \frac{0.1571}{0.05} \times 3 = 9.4286\ A \tag{3}$$

MS320F2800137 和 MSPM01507 子板都可以由该外部 OCP 电路触发。

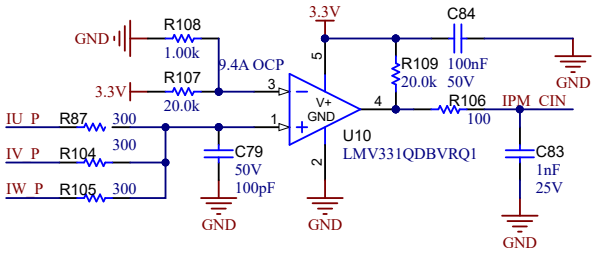


图 2-6. 外部过流保护电路

2.4.1.7 TMS320F2800F137 的内部过流保护

TMS320F2800F137 具有可配置为监控三相电流的内部窗口比较器，内部比较器没有任何软件或中断延迟来触发过流以停止 PWM，这可以快速保护外部 IPM 或功率器件。

2.4.2 三相 PMSM 驱动器

永磁同步电机 (PMSM) 具有一个绕线定子、一个永磁转子组件和用于检测转子位置的内部或外部器件。感测器件提供位置反馈以适当地调整定子基准电压的频率和振幅，从而使磁体组件保持旋转。一个内部永磁转子和外部绕组的组合提供低转子惯性、有效散热和电机尺寸减少等优势。

- 同步电机构造：永磁体被牢牢固定在旋转轴上，生成了一个恒定的转子磁通。这个转子磁通通常具有一个恒定的磁通量。当加电时，定子绕组产生一个旋转的电磁场。为了控制旋转磁场，必须控制定子电流。
- 根据机器的功率范围和额定速度，转子的实际结构会有所不同。永磁体非常适合范围高达几千瓦的同步电机。为了获得更高的额定功率，转子通常由接通直流电的绕组组成。转子的机械结构是针对所需磁极的数量和所需的磁通梯度进行设计的。
- 定子和转子磁通的交感产生了一个转矩。由于定子被牢固地安装在电机架上，而转子可自由旋转，因此转子的旋转会产生一个有用的机械输出，如图 2-7 所示。
- 必须仔细控制转子磁场和定子磁场间的角度，以产生最大扭矩和实现较高的机电转换效率。为了实现这一目的，在同一转速和扭矩条件下，为了尽可能少地消耗电流，在关闭速度环路后需要使用无传感器算法进行微调。
- 旋转中的定子磁场的频率必须与转子永磁磁场的频率相同，否则转子就会经历快速的正负扭矩交替。这导致扭矩产生效果不佳，并且在机器部件上产生过多的机械抖动、噪声和机械应力。此外，如果转子惯性使转子不能对这些摆动做出响应，那么转子在同步频率上停止转动，并且对静止转子的平均扭矩：零扭矩做出响应。这意味着机器会出现一种称为*牵出*的现象。这也是为什么同步机器不能自启动的原因。
- 转子磁场与定子磁场间的角度必须等于 90º 以获得最高的互转矩产出量。为了产生正确的定子磁场，该同步需要知道转子位置。
- 通过将不同转子相位的输出组合在一起，可将定子磁场设定为任一方向和强度以产生相应的定子磁通。

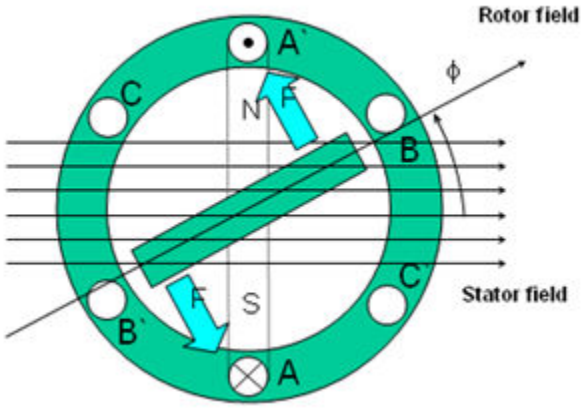


图 2-7. 旋转的定子磁通和转子磁通之间的相互作用产生扭矩

2.4.2.1 PM 同步电机的场定向控制

为了实现更好的动态性能，需要采用更加复杂的控制方案来控制 PM 电机。借助微控制器提供的数学处理能力，可以实施先进的控制策略，这些策略使用数学变换将永磁电机中的扭矩生成和磁化功能解耦。这种解耦的扭矩和磁化控制通常称为转子磁通定向控制，或简称为场定向控制 (FOC)。

在直流 (DC) 电机中，定子和转子的励磁是独立控制的，产生的扭矩和磁通可以独立调整，如图 2-8 所示。磁场激励强度（例如，磁场激励电流的振幅）决定了磁通的大小。通过转子绕组的电流确定了扭矩是如何生成。转子上的换向器在扭矩产生过程中发挥着有趣的作用。换向器与电刷接触，这个机械构造旨在将电路切换至机械对齐的绕组以产生最大的扭矩。这样的安排意味着，电机的扭矩产生在任何时候都非常接近于理想情况。这里的关键点是，通过管理绕组以保持转子绕组产生的磁通与定子磁场垂直。

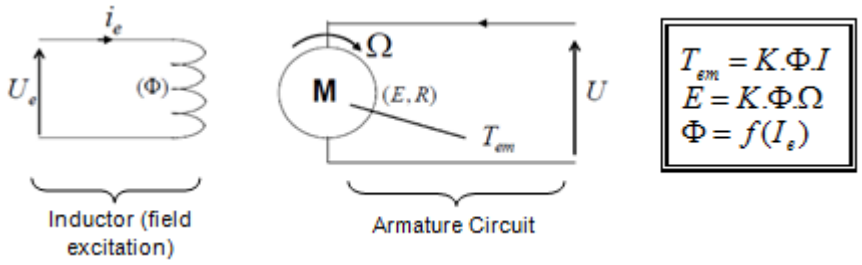


图 2-8. 在直流电机模型中磁通和扭矩是独立控制的

同步和异步电机上 FOC（也被称为矢量控制）的目的在于能够分别控制转矩生成和磁化磁通分量。FOC 控制可实现转矩和定子电流磁化磁通分量的去耦。借助于磁化的去耦合控制，定子磁通的扭矩生成分量现在可以被看成是独立扭矩控制。为了去耦合扭矩和磁通，有必要采用几个数学变换，而这是最能体现微控制器价值的地方。微控制器提供的处理能力可非常快速地执行使这些数学变换。反过来，这意味着控制电机的整个算法可以高速率执行，从而实现了更高的动态性能。除了去耦合，现在一个电机的动态模型被用于很多数量的计算，例如转子磁通角和转子速度。这意味着，它们的影响被计算在内，并且总体控制质量更佳。

根据电磁定律，同步电机中产生的扭矩等于两个现有磁场的矢量叉积，如方程式 4 所示。

$$\tau_{em} = \vec{B}_{stator} \times \vec{B}_{rotor} \tag{4}$$

该表达式表明，如果定子和转子磁场*正交*，则扭矩最大，这意味着我们需要将负载保持在 90 度。如果始终可以提供此条件且磁通可以正确定向，则会降低扭矩纹波并提供更好的动态响应。然而，您需要了解转子的位置：这可以通过位置传感器（诸如递增编码器）实现。对于无法接近转子的低成本应用，采用不同的转子位置观察器策略可无需使用位置传感器。

简而言之，目标是使转子和定子磁通保持正交：例如，目标是将定子磁通与转子磁通的 q 轴对齐，从而与转子磁通正交。为了实现这个目的，控制与转子磁通正交的定子电流分量以产生命令规定的扭矩，并且直接分量被设定为零。定子电流的直接分量可用在某些磁场减弱的情况下，这有抗拒转子磁通的作用，并且减少反电动势，从而实现更高速的运行。

FOC 包含控制由一个矢量表示的定子电流。这个控制所基于的设计是，将三相时间和速度相关系统变换为两坐标（d 和 q 坐标）非时变系统。这些设计导致一个与 DC 机器控制结构相似的结构。FOC 机器需要两个常数作为输入基准：转矩分量（与 q 坐标对其）和磁通分量（与 d 坐标对其）。由于 FOC 完全基于这些设计，此控制结构处理即时电量。这使得在每次的工作运转过程中（稳定状态和瞬态）均可实现准确控制，并且与受限带宽数学模型无关。因此，FOC 通过以下方式解决了传统方案存在的问题：

- 轻松达到恒定基准（定子电流的扭矩分量和磁通分量）
- 轻松应用直接扭矩控制，这是因为在 (d, q) 坐标系中，扭矩的表达式定义如方程式 5 所示。

$$\tau_{em} \propto \psi_R \times i_{sq} \tag{5}$$

通过将转子磁通 (ψ_R) 的振幅保持在一个固定值，在扭矩和扭矩分量 (i_{Sq}) 之间实现线性关系。因此，可通过控制定子电流矢量的转矩分量来控制转矩。

2.4.2.1.1 空间矢量定义和投影

交流电机的三相电压、电流和磁通可根据复数空间矢量进行分析。对于电流，空间矢量可定义如下。假设 i_a、i_b、i_c 是定子的三相即时电流，则复数定子电流矢量的定义如方程式 6 所示。

$$\bar{i}_s = i_a + \alpha i_b + \alpha^2 i_c \tag{6}$$

其中

- $\alpha = e^{j\frac{2}{3}\pi}$ 和 $\alpha^2 = e^{j\frac{4}{3}\pi}$ 表示空间运算元

图 2-9 所示为定子电流的复数空间矢量。

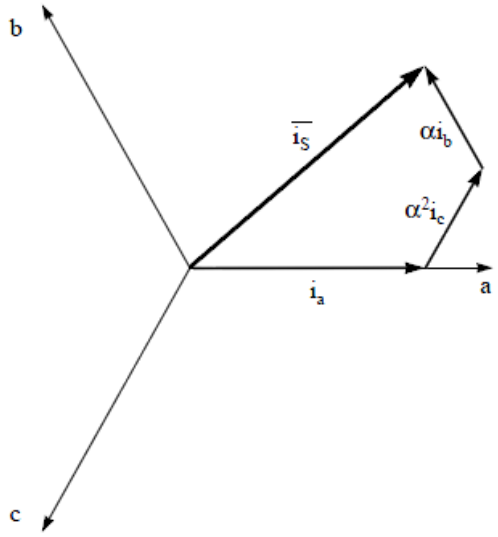


图 2-9. 定子电流空间矢量及其以 (a,b,c) 坐标系表示的分量

其中

- a、b 和 c 是三相系统轴

这个电流空间矢量描述了三相正弦系统，这个系统仍然需要变换为一个两坐标非时变系统。这个变换可拆分为两个步骤：

- (a, b) ⇒ (α, β)（Clarke 变换），输出一个两坐标时变系统
- (α, β) ⇒ (d, q)（Park 变换），输出一个两坐标时不变系统

2.4.2.1.1.1 (a, b) ⇒ (α, β) *Clarke 变换*

可以使用另外一个仅包含两相 (α，β) 正交轴的坐标系来表示该空间矢量。假设 a 轴和 α 轴方向相同，可以得到图 2-10 所示的矢量图。

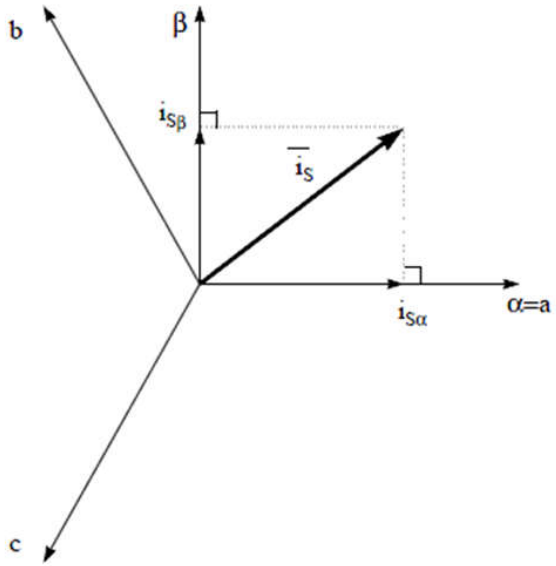


图 2-10. 静止坐标系中的定子电流空间矢量

将三相系统修改为 (α , β) 二维正交系统的投影如方程式 7 所示。

$$i_{s\alpha} = i_a$$
$$i_{s\beta} = \frac{1}{\sqrt{3}} i_a + \frac{2}{\sqrt{3}} i_b \quad (7)$$

两相 (α , β) 电流仍取决于时间和速度。

2.4.2.1.1.2 $(\alpha, \beta) \Rightarrow (d, q)$ Park 变换

这是 FOC 内最重要的变换。事实上，该投影在 (d, q) 旋转坐标系中修改了一个两相正交系统 (α , β)。假设 d 轴与转子磁通对齐，那么图 2-11 显示了来自该二维坐标系的电流矢量的关系。

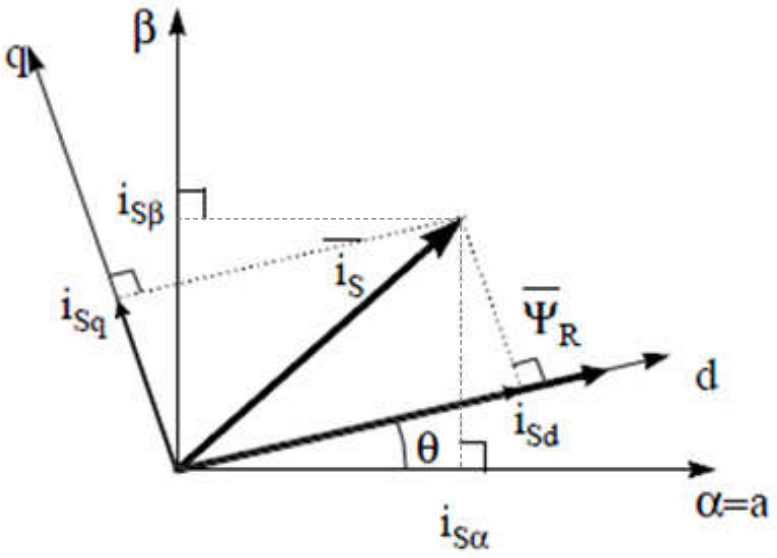


图 2-11. d,q 旋转坐标系中的定子电流空间矢量

电流矢量的磁通和扭矩分量由方程式 8 决定。

$$i_{sd} = i_{s\alpha}\cos(\theta) + i_{s\beta}\sin(\theta)$$
$$i_{sq} = -i_{s\alpha}\sin(\theta) + i_{s\beta}\cos(\theta) \quad (8)$$

其中

- θ 是转子磁通位置

这些分量取决于电流矢量 (α , β) 分量和转子磁通位置；如果知道正确的转子磁通位置，那么，通过该投影，d,q 分量就变成一个常量。现在，两个相位电流变换为直流数量 (非时变)。此时扭矩控制变得更容易，其中恒定的 i_{sd} (磁通分量) 和 i_{sq} (扭矩分量) 电流分量单独受到控制。

2.4.2.1.2 交流电机 FOC 基本配置方案

图 2-12 总结了用 FOC 进行扭矩控制的基本系统配置。

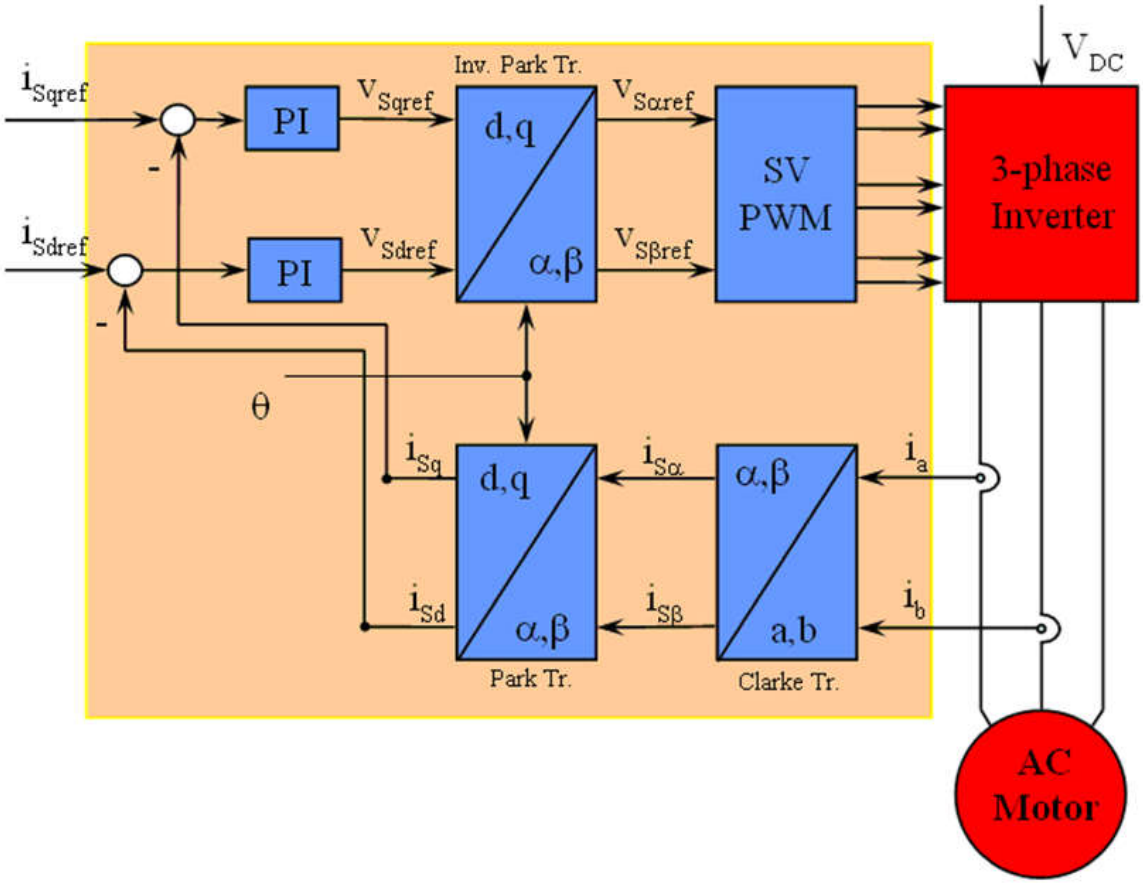


图 2-12. 交流电机 FOC 基本配置方案

测量了两个电机相电流。这些测量值馈入 Clarke 变换模块。这个模块的输出为 $i_{s\alpha}$ 和 $i_{s\beta}$。电流的这两个分量是 Park 变换的输入，该变换给出了 d,q 旋转坐标系中的电流。i_{sd} 和 i_{sq} 分量与基准 i_{sdref}（磁通基准分量）和 i_{sqref}（扭矩基准分量）进行比较。此时，这个控制结构具有一个有意思的优势：只需改变磁通基准并获得转子磁通位置，该结构即可用于控制同步或感应电机。与在同步永磁电机中一样，转子磁通是固定的，并由磁体确定；所以无需产生转子磁通。因此，当控制 PMSM 时，将 i_{sdref} 设置为零。由于交流感应电机需要生成转子磁通才能运行，因此磁通基准一定不能为零。这很方便地解决了*经典*控制结构的一个主要缺陷：异步驱动至同步驱动的可移植性。当使用转速 FOC 时，扭矩命令 i_{sqref} 可以是转速调节器的输出。电流调节器的输出是 V_{sdref} 和 V_{sqref}；这些输出应用于 Park 逆变换。这个模块的输出是 $V_{s\alpha ref}$ 和 $V_{s\beta ref}$，它们是（α，β）静止正交坐标系中定子矢量电压的分量。这些是空间矢量脉宽调制 (PWM) 的输入。这个块的输出是驱动此反相器的信号。请注意，Park 和 Park 逆变换均需要转子磁通位置。这个转子磁通位置的获得由交流机器的类型（同步或异步机器）而定。

2.4.2.1.3 转子磁通位置

转子磁通位置的相关知识是 FOC 的核心。事实上，如果该变量存在误差，则转子磁通与 d 轴不对齐，并且定子电流的磁通和扭矩分量 i_{sd} 和 i_{sq} 不正确。图 2-13 展示了 (a, b, c)、(α，β) 和 (*d, q*) 坐标系，以及转子磁通的正确位置和以同步速度随 (d, q) 坐标旋转的定子电流和定子电压空间矢量。

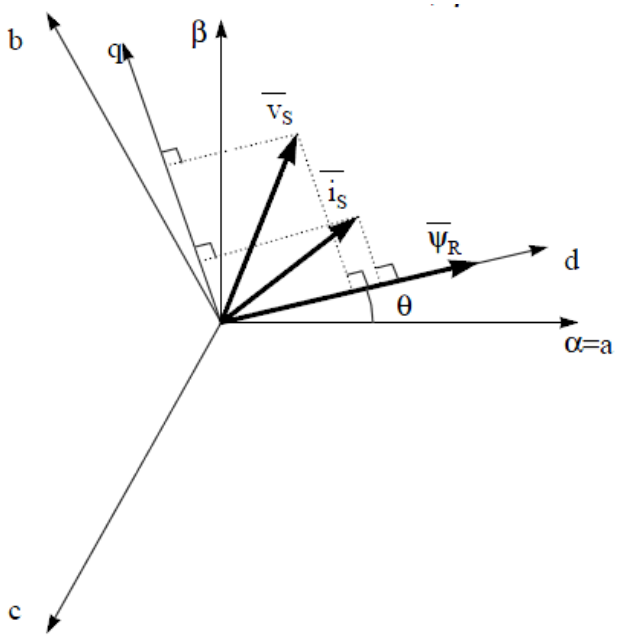


图 2-13. (d, q) 旋转坐标系中的电流、电压和转子磁通空间矢量

考虑同步或异步电机，转子磁通位置的测量是不同的：

- 在同步电机中，转子转速等于转子磁通转速。然后 θ（转子磁通位置）由位置传感器或转子速度的积分直接计算。
- 在异步电机中，转子转速不等于转子磁通转速（存在转差速度），因此需要使用特定的方法来计算 θ。基本方法是使用一个电流模型，该模型需要 *d, q* 坐标系中的电机模型的两个公式。

理论上，针对 PMSM 驱动的 FOC 可用磁通实现对电机转矩的单独控制，这与直流电机的运行类似。换句话说，转矩和磁通互相之间去耦合。从静止基准框架到同步旋转基准框架间的变量变换需要知道转子位置信息。由于这种变换（所谓的 Park 变换），q 轴电流控制扭矩，而 d 轴电流强制设置为零。因此，该系统的关键模块是使用增强型滑模观测器 (eSMO) 或 FAST 估算器来估算转子位置。

图 2-14 展示了该参考设计中风扇 PMSM 的无传感器 FOC（使用 eSMO 并具有快速启动功能）的整体方框图。

图 2-15 展示了该参考设计中压缩机 PMSM 的无传感器 FOC（使用 eSMO 并具有弱磁控制 (FWC) 和每安培最大扭矩 (MTPA) 功能）的整体方框图。

图 2-16 展示了该参考设计中风扇 PMSM 的无传感器 FOC（使用 FAST 并具有快速启动功能）的整体方框图。

图 2-17 展示了该参考设计中压缩机 PMSM 的无传感器 FOC（使用 FAST 并具有弱磁控制 (FWC) 和每安培最大扭矩 (MTPA) 功能）的整体方框图。

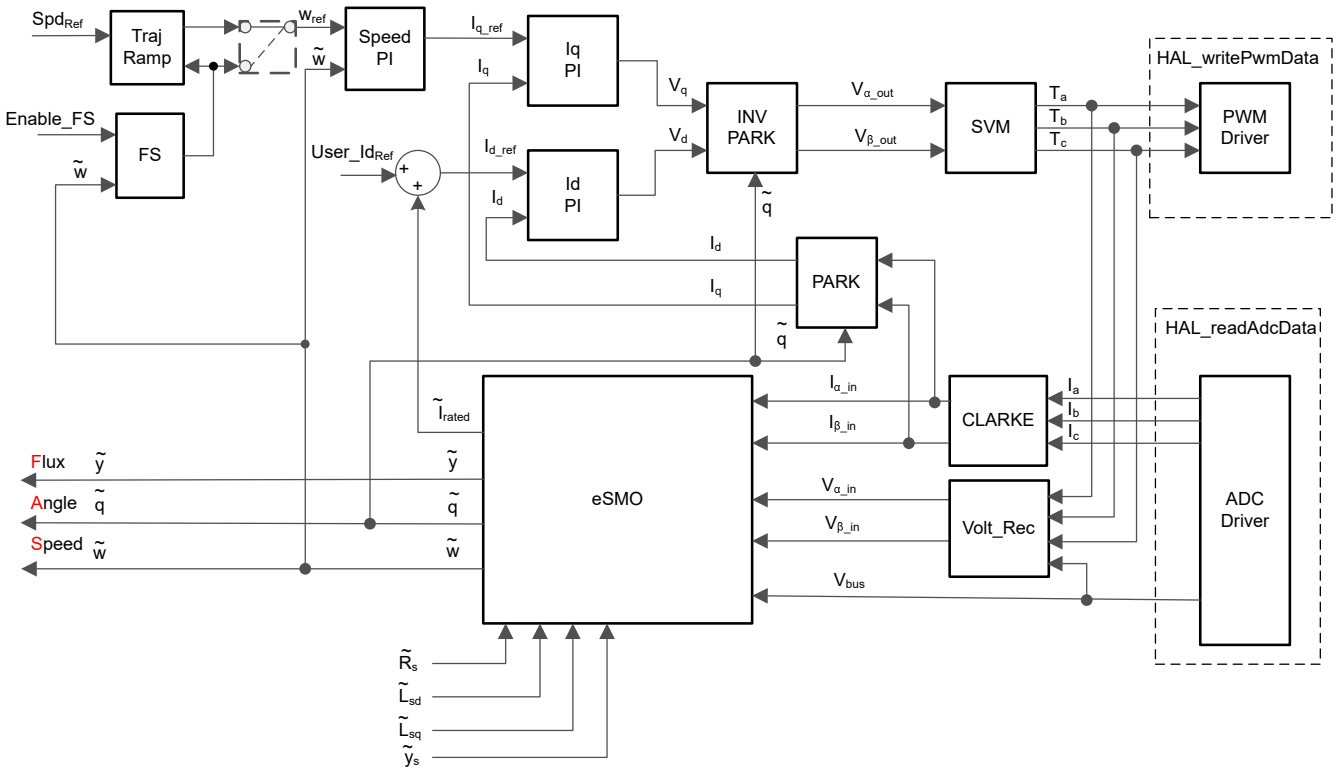


图 2-14. 使用 eSMO 并具有快速启动 (FS) 功能的 PMSM 的无传感器 FOC

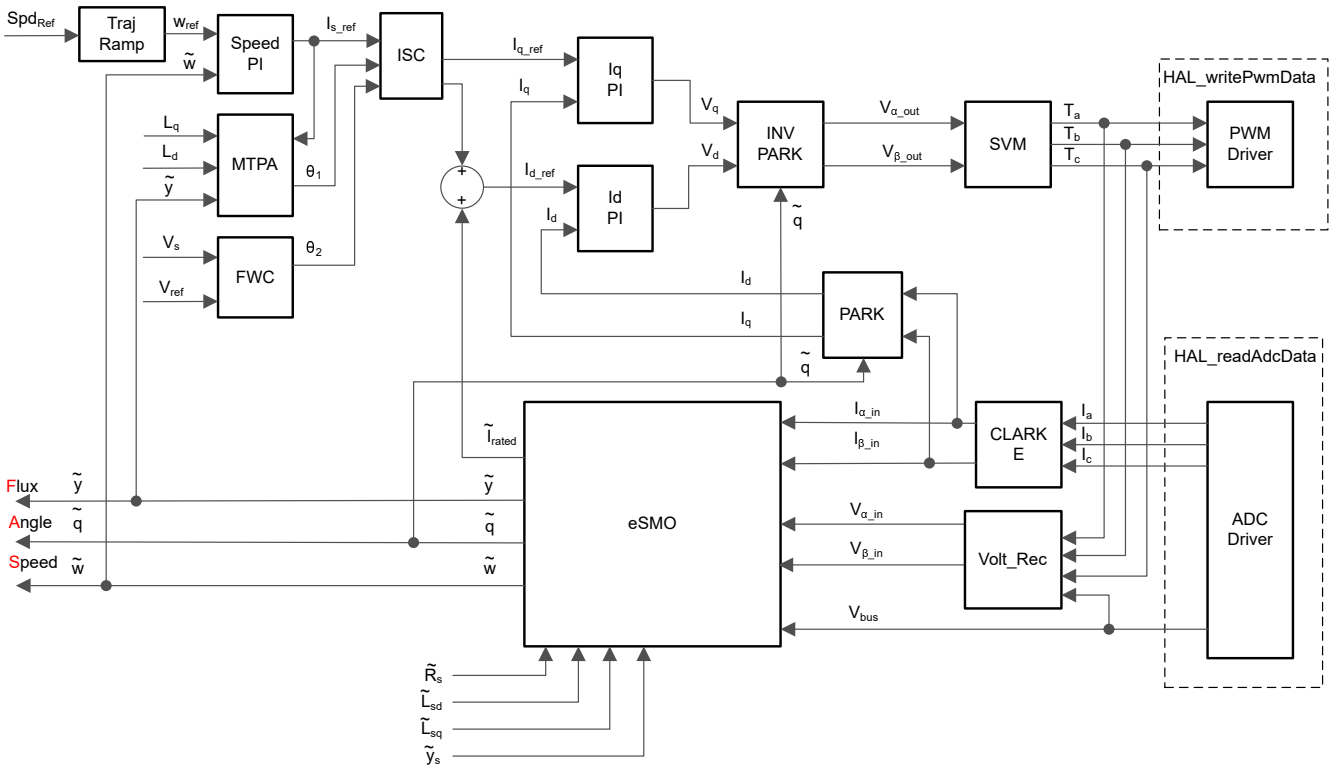


图 2-15. 使用 eSMO 并具有 FWC 和 MTPA 功能的 PMSM 的无传感器 FOC

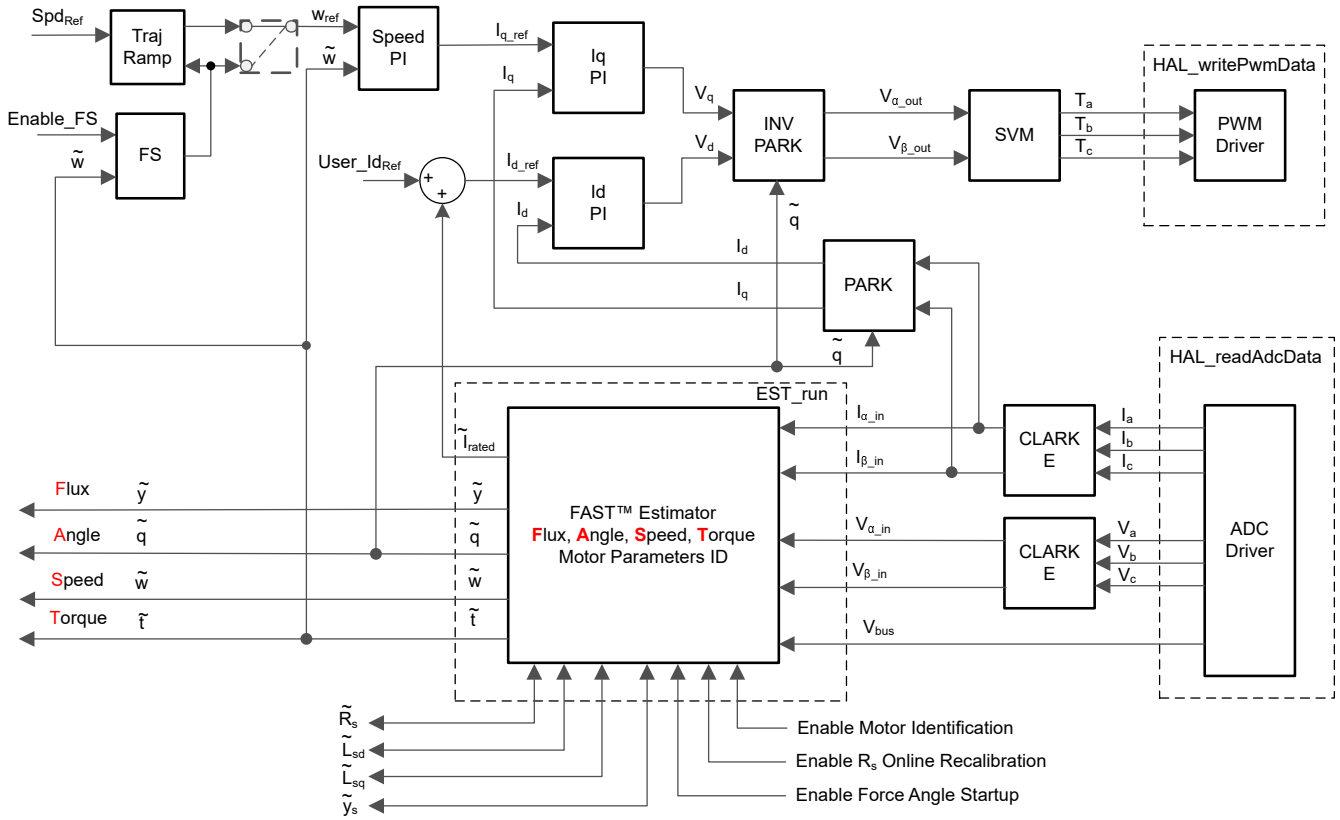


图 2-16. 使用 FAST 并具有快速启动 (FS) 功能的 PMSM 的无传感器 FOC

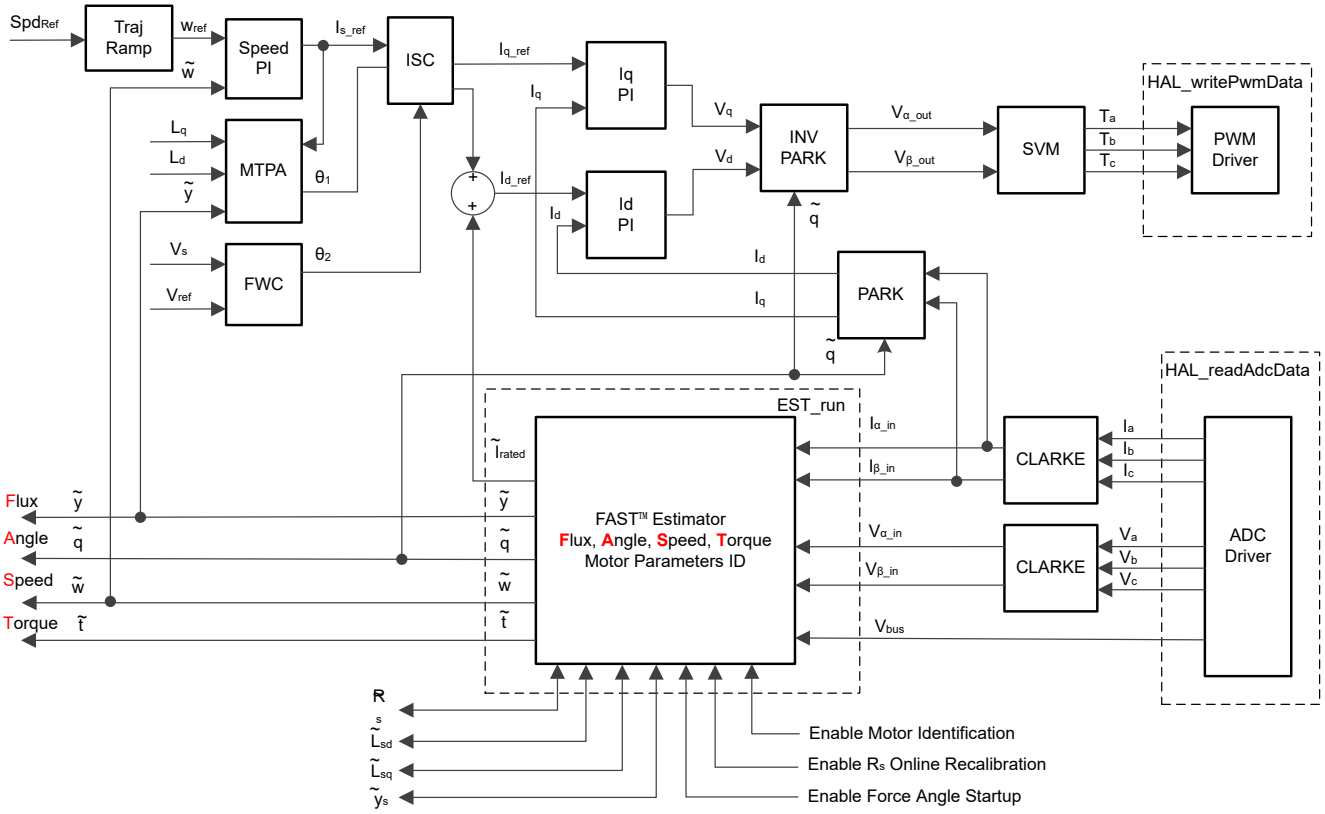


图 2-17. 使用 FAST 并具有 FWC 和 MTPA 功能的 PMSM 的无传感器 FOC

2.4.2.2 PM 同步电机的无传感器控制

在家用电器应用中，使用机械传感器会增加成本、尺寸和可靠性问题。为了克服这些问题，无传感器控制方法应运而生，可以通过多种估算方法在没有机械位置传感器的情况下获得转子转速和位置信息。滑模观测器 (SMO) 因其各种吸引人的特性（包括可靠性、所需的性能和针对系统参数变化的稳健性）而被广泛使用。

2.4.2.2.1 具有锁相环的增强型滑模观测器

基于模型的方法用于实现 IPMSM 驱动系统在电机以中高速运行时的无位置传感器控制。模型法通过反电动势或磁链模型估算转子位置。滑动模式观测器是基于滑模控制的观测器设计方法。系统的结构不是固定的，而是根据系统的当前状态有目的地改变，迫使系统按照预定的滑模轨迹运动。其优点包括响应速度快、稳健性高以及对参数变化和干扰不敏感。

2.4.2.2.1.1 IPMSM 的数学模型和 FOC 结构

IPMSM 的无传感器 FOC 结构如图 2-18 所示。在该系统中，eSMO 用于实现 IPMSM 系统的无传感器控制，eSMO 模型是利用反电动势模型和 PLL 模型设计的，用于估算转子位置和转速。

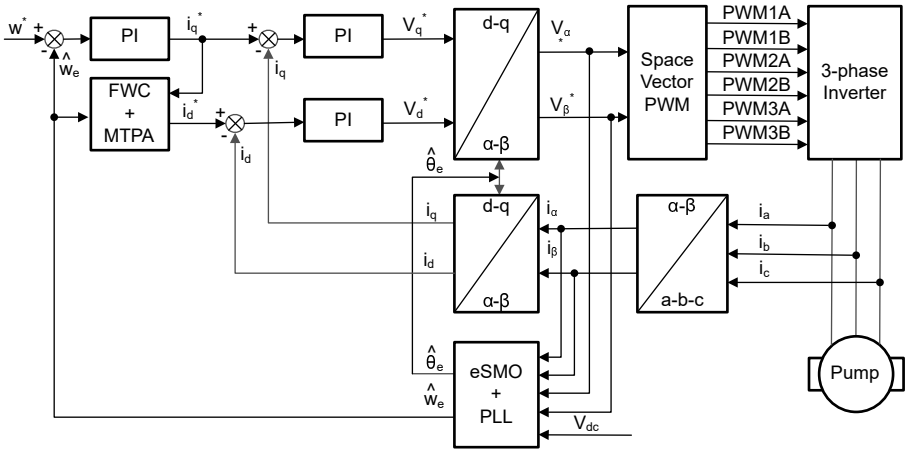


图 2-18. IPMSM 系统的无传感器 FOC 结构

IPMSM 由一个三相定子绕组（a、b、c 轴）和用于励磁的永磁体 (PM) 转子组成。电机由标准的三相逆变器进行控制。可以使用相位 a-b-c 量对 IPMSM 进行建模。通过适当的坐标变换，可以得到 d-q 转子坐标系和 α - β 静止坐标系中的动态 PMSM 模型。这些坐标系之间的关系如方程式 9 所示。通用 PMSM 的动态模型可以在 d-q 转子坐标系中写为：

$$\begin{bmatrix} v_d \\ v_q \end{bmatrix} = \begin{bmatrix} R_s + pL_d & -\omega_e L_q \\ \omega_e L_d & R_s + pL_q \end{bmatrix} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_e \lambda_{pm} \end{bmatrix} \quad (9)$$

其中

- v_d 和 v_q 分别是 q 轴和 d 轴定子端电压
- i_d 和 i_q 分别是 d 轴和 q 轴定子电流
- L_d 和 L_q 分别是 q 轴和 d 轴电感
- *P* 是导数算子，用于简写 $\frac{d}{dt}$
- λ_{pm} 是永磁体产生的磁链
- R_s 是定子绕组的电阻
- ω_e 是转子的电角速度

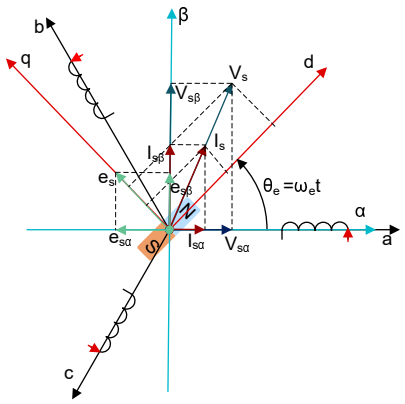


图 2-19. PMSM 建模坐标系的定义

通过使用如图 2-19 所示的 Park 逆变换，PMSM 的动力学可以在 α - β 静止坐标系中按照方程式 10 所示进行建模：

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} R_s + pL_d & \omega_e(L_d - L_q) \\ -\omega_e(L_d - L_q) & R_s + pL_q \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \begin{bmatrix} e_\alpha \\ e_\beta \end{bmatrix} \tag{10}$$

其中

- e_α 和 e_β 是 α - β 轴上扩展电动势 (EEMF) 的分量，其定义如方程式 11 中所示：

$$\begin{bmatrix} e_\alpha \\ e_\beta \end{bmatrix} = \left(\lambda_{pm} + (L_d - L_q) i_d\right) \omega_e \begin{bmatrix} -\sin(\theta_e) \\ \cos(\theta_e) \end{bmatrix} \tag{11}$$

根据方程式 10 和方程式 11，通过等效变换和引入 EEMF 概念，可以将转子位置信息从电感矩阵中解耦出来，从而使 EEMF 成为唯一包含转子磁极位置信息的项。然后可以直接利用 EEMF 相位信息实现转子位置观测。使用定子电流作为状态变量，将 IPMSM 电压公式方程式 10 改写为状态公式：

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \frac{1}{L_d} \begin{bmatrix} -R_s & -\omega_e(L_d - L_q) \\ \omega_e(L_d - L_q) & -R_s \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \frac{1}{L_d} \begin{bmatrix} V_\alpha - e_\alpha \\ V_\beta - e_\beta \end{bmatrix} \tag{12}$$

由于定子电流是唯一可以直接测量的物理量，因此在定子电流路径上选择滑动面：

$$S(x) = \begin{bmatrix} \hat{i}_\alpha - i_\alpha \\ \hat{i}_\beta - i_\beta \end{bmatrix} = \begin{bmatrix} \tilde{i}_\alpha \\ \tilde{i}_\beta \end{bmatrix} \tag{13}$$

其中

- $\hat{i}_\alpha$ 和 $\hat{i}_\beta$ 是估算的电流
- 上标 **^** 表示变量为估算值
- 上标 “˜” 表示变量为变量误差，即观测值与实际测量值之间的差异

2.4.2.2.1.2 IPMSM 的 ESMO 设计

图 2-20 展示了集成到 SMO 中的传统 PLL。

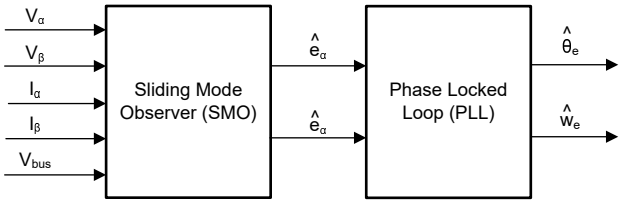


图 2-20. 包含用于 PMSM 的 PLL 的 eSMO 方框图

这里构建了传统的降阶滑模观测器，其数学模型如方程式 14 所示，方框图如图 2-21 所示。

$$\begin{bmatrix} \hat{i}_\alpha \\ \hat{i}_\beta \end{bmatrix} = \frac{1}{L_d}\begin{bmatrix} -R_s & -\hat{\omega}_e(L_d - L_q) \\ \hat{\omega}_e(L_d - L_q) & -R_s \end{bmatrix}\begin{bmatrix} \hat{i}_\alpha \\ \hat{i}_\beta \end{bmatrix} + \frac{1}{L_d}\begin{bmatrix} V_\alpha - \hat{e}_\alpha + z_\alpha \\ V_\beta - \hat{e}_\beta + z_\beta \end{bmatrix} \quad (14)$$

其中

- z_α 和 z_β 是滑模反馈分量，其定义如方程式 15 所示：

$$\begin{bmatrix} z_\alpha \\ z_\beta \end{bmatrix} = \begin{bmatrix} k_\alpha \text{sign}(\hat{i}_\alpha - i_\alpha) \\ k_\beta \text{sign}(\hat{i}_\beta - i_\beta) \end{bmatrix} \quad (15)$$

其中

- k_α 和 k_β 是通过李雅普诺夫稳定性分析设计的恒定滑模增益

如果 k_α 和 k_β 是足够大的正值，以提供 SMO 的稳定运行，然后 k_α 和 k_β 足够大，以保持 $k_\alpha > \max(|e_\alpha|)$ 和 $k_\beta > \max(|e_\beta|)$。

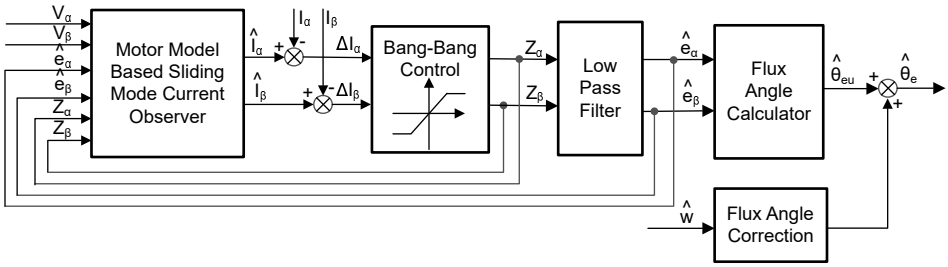


图 2-21. 传统滑模观测器的方框图

α - β 轴上的 EEMF 估算值 ($\hat{e}_\alpha$, $\hat{e}_\beta$) 可通过低通滤波器从不连续开关信号中获得，这些信号为 z_α 和 z_α：

$$\begin{bmatrix} \hat{e}_\alpha \\ \hat{e}_\beta \end{bmatrix} = \frac{\omega_c}{s + \omega_c}\begin{bmatrix} z_\alpha \\ z_\beta \end{bmatrix} \quad (16)$$

其中

- $\omega_c = 2\pi f_c$ 是 LPF 的截止角频率，通常根据定子电流的基频来选择该截止角频率

因此，转子位置可以直接通过反电动势的反正切计算得出，其定义如方程式 17 所示：

$$\hat{\theta}_e = -\tan^{-1}\left(\frac{\hat{e}_\alpha}{\hat{e}_\beta}\right) \quad (17)$$

低通滤波器消除了滑模函数的高频项，从而导致出现相位延迟。可以通过截止频率 ω_c 和反电动势频率 ω_e 之间的关系对其进行补偿，定义如方程式 18 所示：

$$\Delta\theta_e = -\tan^{-1}\left(\frac{\omega_e}{\omega_c}\right) \tag{18}$$

这样使用 SMO 方法估算的转子位置就如方程式 19 所示：

$$\hat{\theta}_e = -\tan^{-1}\left(\frac{\hat{e}_\alpha}{\hat{e}_\beta}\right) + \Delta\theta_e \tag{19}$$

在数字控制应用中，需要使用 SMO 的时间离散方程。欧拉法是变换为时间离散观测器的合适方法。在 α - β 坐标中，方程式 14 的时间离散系统矩阵由方程式 20 给出：

$$\begin{bmatrix}\hat{i}_\alpha(n+1)\\ \hat{i}_\beta(n+1)\end{bmatrix} = \begin{bmatrix}F_\alpha\\ F_\beta\end{bmatrix}\begin{bmatrix}\hat{i}_\alpha(n)\\ \hat{i}_\beta(n)\end{bmatrix} + \begin{bmatrix}G_\alpha\\ G_\beta\end{bmatrix}\begin{bmatrix}V^*_\alpha(n) - \hat{e}_\alpha(n) + z_\alpha(n)\\ V^*_\beta(n) - \hat{e}_\beta(n) + z_\beta(n)\end{bmatrix} \tag{20}$$

其中

- 矩阵 [F] 益 [G] 由方程式 21 和方程式 22 给出：

$$\begin{bmatrix}F_\alpha\\ F_\beta\end{bmatrix} = \begin{bmatrix}e^{-\frac{R_s}{L_d}}\\ e^{-\frac{R_s}{L_q}}\end{bmatrix} \tag{21}$$

$$\begin{bmatrix}G_\alpha\\ G_\beta\end{bmatrix} = \frac{1}{R_s}\begin{bmatrix}1 - e^{-\frac{R_s}{L_d}}\\ 1 - e^{-\frac{R_s}{L_q}}\end{bmatrix} \tag{22}$$

方程式 16 的时间离散形式由方程式 23 给出：

$$\begin{bmatrix}\hat{e}_\alpha(n+1)\\ \hat{e}_\beta(n+1)\end{bmatrix} = \begin{bmatrix}\hat{e}_\alpha(n)\\ \hat{e}_\beta(n)\end{bmatrix} + 2\pi f_c\begin{bmatrix}z_\alpha(n) - \hat{e}_\alpha(n)\\ z_\beta(n) - \hat{e}_\beta(n)\end{bmatrix} \tag{23}$$

2.4.2.2.1.3 使用 PLL 的转子位置和转速估算

在反正切法中，由于噪声和谐波分量的存在，位置和转速估算的精度会受到影响。为了消除该问题，可使用 PLL 模型对 IPMSM 的无传感器控制结构中的转速和位置进行估算。节 2.4.2.2.1.2 中说明了与 SMO 配合使用的 PLL 结构。反电动势估算 $\hat{e}_\alpha$ 和 $\hat{e}_\beta$ 可与 PLL 模型配合使用来估算电机角速度和位置，如图 2-22 所示。

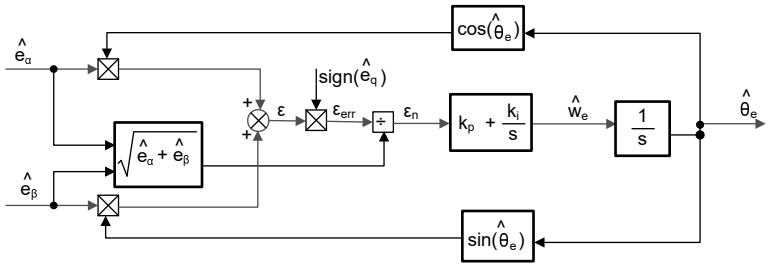


图 2-22. 锁相环位置跟踪器的方框图

由于 $e_\alpha = E\cos(\theta_e)$，$e_\beta = E\sin(\theta_e)$ 以及 $E = \omega_e\lambda_{pm}$，位置误差定义如方程式 24 所示：

$$\varepsilon = \hat{e}_\beta\cos(\hat{\theta}_e) - \hat{e}_\alpha\sin(\hat{\theta}_e) = E\sin(\theta_e)\cos(\hat{\theta}_e) - E\cos(\theta_e)\sin(\hat{\theta}_e) = E\sin(\theta_e - \hat{\theta}_e) \tag{24}$$

其中

- E 是 EEMF 的幅度，与电机转速成正比 ω_e

当 $\left(\theta_e - \hat{\theta}_e\right) < \frac{\pi}{2}$ ，然后方程式 25 可简化为方程式 24：

$$\varepsilon = E\left(\theta_e - \hat{\theta}_e\right) \tag{25}$$

可以进一步得到 EEMF 归一化后的位置误差（方程式 26）：

$$\varepsilon_n = \theta_e - \hat{\theta}_e \tag{26}$$

根据分析，可以得到正交锁相环位置跟踪器的简化方框图，如图 2-23 所示。PLL 的闭环传递函数可表示为方程式 27：

$$\frac{\hat{\theta}_e}{\theta_e} = \frac{k_p s + k_i}{s^2 + k_p s + k_i} = \frac{2\xi\omega_n s + \omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{27}$$

其中

- k_p 和 k_i 是标准 PI 调节器的比例增益和积分增益

固有频率 ω_n 和阻尼比 ξ 由方程式 28 中给出：

$$k_p = 2\xi\omega_n : \quad k_i = \omega_n^2 \tag{28}$$

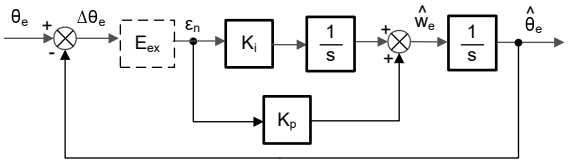


图 2-23. 锁相环位置跟踪器的简化方框图

2.4.2.3 弱磁 (FW) 和每安培最大扭矩 (MTPA) 控制

永磁同步电机 (PMSM) 因其高功率密度、高效率和宽转速范围而广泛应用于家用电器应用。PMSM 包含两种主要类型：表面贴装式 PMSM (SPM) 和内嵌式 PMSM (IPM)。由于 SPM 电机在扭矩和 q 轴电流之间具有线性关系，因此更易于控制。不过，IPMSM 由于凸极比大而具有电磁扭矩和磁阻扭矩。总扭矩相对于转子角度是非线性的。因此，MTPA 技术可用于 IPM 电机，以优化恒定扭矩区域中的扭矩生成。弱磁控制的目的是优化以达到 PMSM 驱动器的最高功率和效率。弱磁控制可以使电机以其基本转速运行，扩大其运行限值以使转速高于额定转速，并允许在整个转速和电压范围内实现最佳控制。

IPMSM 数学模型的电压公式可以用 d-q 坐标来描述，如方程式 29 和方程式 30 所示。

$$v_d = L_d\frac{di_d}{dt} + R_s i_d - p\omega_m L_q i_q \tag{29}$$

$$v_q = L_q\frac{di_q}{dt} + R_s i_q + p\omega_m L_d i_d + p\omega_m \psi_m \tag{30}$$

图 2-24 展示了 IPM 同步电机的动态等效电路。

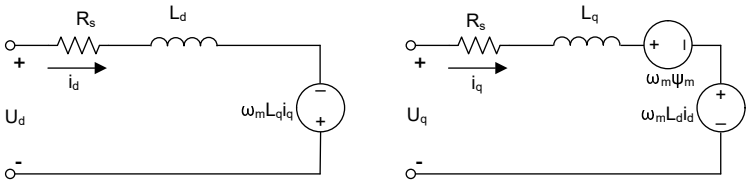


图 2-24. IPM 同步电机的等效电路

IPMSM 产生的总电磁扭矩可以由方程式 31 表示，产生的扭矩包含两个不同的项。第一项对应于扭矩电流 i_q 和永磁体 ψ_m 之间产生的相互反作用力扭矩，而第二项对应于由于 d 轴和 q 轴上的电感不同而产生的磁阻扭矩。

$$T_e = \frac{3}{2}p\left[\psi_m i_q + \left(L_d - L_q\right)i_d i_q\right] \tag{31}$$

在大多数应用中，IPMSM 驱动器具有转速和扭矩约束，这主要是由于分别存在逆变器或电机额定电流以及可用的直流链路电压限制。这些约束可以用数学公式方程式 32 和方程式 33 表示。

$$I_a = \sqrt{i_d^2 + i_q^2} \le I_{max} \tag{32}$$

$$V_a = \sqrt{v_d^2 + v_q^2} \le V_{max} \tag{33}$$

其中

- V_{max} 和 I_{max} 是逆变器或电机允许的最大电压和电流

在两级三相电压源逆变器 (VSI) 供电的电机中，可实现的最大相电压受直流链路电压和 PWM 策略的限制。如果采用空间矢量调制 (SVPWM)，则最大电压限制为方程式 34 中所示的值。

$$\sqrt{v_d^2 + v_q^2} \le v_{max} = \frac{v_{dc}}{\sqrt{3}} \tag{34}$$

通常，定子电阻 R_s 在高速运行时可以忽略不计，并且电流的导数在稳态下为零，因此得到方程式 35，如下所示。

$$\sqrt{L_d^2\left(i_d + \frac{\psi_{pm}}{L_d}\right)^2 + L_q^{2,2} i_q^2} \le \frac{V_{max}}{\omega_m} \tag{35}$$

方程式 32 的电流限制在 d-q 平面中产生一个半径为 I_{max} 的圆，而方程式 34 的电压限制产生一个椭圆，其半径 V_{max} 随着转速的增加而减小。必须对得到的 d-q 平面电流矢量进行控制，使其同时遵守电流和电压约束。根据这些约束，可以区分 IPMSM 的三个工作区域，如图 2-25 所示。

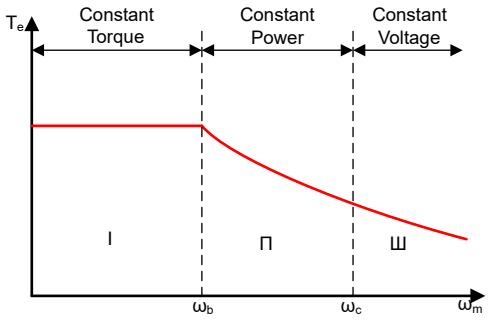


图 2-25. IPMSM 控制工作区域

1. 恒定扭矩区域：可以在该工作区域内实施 MTPA，从而可产生最大扭矩。
2. 恒定功率区域：必须采用弱磁控制，并且在达到电流约束时减小扭矩容量。
3. 恒定电压区域：在这个工作区域，深度弱磁控制使定子电压保持恒定，以尽可能大地产生扭矩。

在恒定扭矩区域，根据方程式 31，IPMSM 的总扭矩包括来自磁链的电磁扭矩和来自以下电感之间凸极的磁阻扭矩： L_d 和 L_q。电磁扭矩与 q 轴电流 i_q 成正比，磁阻扭矩与 d 轴电流 i_d、q 轴电流 i_q 以及 L_d 和 L_q。

SPM 电机的传统矢量控制系统仅通过将命令的 i_d 设置为零来实现非弱磁模式，从而利用电磁扭矩。但是，虽然 IPMSM 利用电机的磁阻扭矩，设计人员还必须控制 d 轴电流。MTPA 控制的目的是计算基准电流 i_d 和 i_q 以尽可能增大产生的电磁扭矩与磁阻扭矩之间的比率。以下各公式显示了 i_d 和 i_q 之间的关系以及定子电流 I_s 的矢量和。

$$I_s = \sqrt{i_d^2 + i_q^2} \tag{36}$$

$$I_d = I_s \cos\beta \tag{37}$$

$$I_q = I_s \sin\beta \tag{38}$$

其中

- β 是同步 (d-q) 坐标系中的定子电流角度

方程式 31 可以表示为方程式 39，其中 I_s 替换了 i_d 和 i_q 。

方程式 39 表明电机扭矩取决于定子电流矢量的角度：

$$T_e = \frac{3}{2} p I_s \sin\beta \left[\psi_m + (L_d - L_q) I_s \cos\beta \right] \tag{39}$$

该公式显示，当电机扭矩微分等于零时，可以计算出最大效率点。当该微分 $\frac{dT_e}{d\beta}$ 为零（如方程式 40 所示）时，可以找到 MTPA 点。

$$\frac{dT_e}{d\beta} = \frac{3}{2} p \left[\psi_m I_s \cos\beta + (L_d - L_q) I_s^2 \cos 2\beta \right] = 0 \tag{40}$$

根据这个公式，可以通过方程式 41 得出 MTPA 控制的电流角度。

$$\beta_{mtpa} = \cos^{-1} \frac{-\psi_m + \sqrt{\psi_m^2 + 8 \times (L_d - L_q)^2 \times I_s^2}}{4 \times (L_d - L_q) \times I_s} \tag{41}$$

因此，可以使用 MTPA 控制的电流角度通过方程式 42 和方程式 43 来表示有效的 d 轴和 q 轴基准电流。

$$I_d = I_s \times \cos\beta_{mtpa} \tag{42}$$

$$I_q = I_s \times \sin\beta_{mtpa} \tag{43}$$

不过，如方程式 41 所示，MTPA 控制的角度 β_{mtpa} 与 d 轴和 q 轴电感有关。这意味着电感的变化会阻碍找到出色的 MTPA 点。为了提高电机驱动器的效率，在线估算 d 轴和 q 轴电感，但参数 L_d 和 L_q 不易于在线测量，并且受饱和效应的影响。稳健的查找表 (LUT) 方法可确保电气参数变化下的可控性。通常，为了简化数学模型，可以忽略 d 轴和 q 轴电感之间的耦合效应。因此，假设 L_d 仅随 i_d 而变化， L_q 仅随 i_q 而变化。因此，d 轴和 q 轴电感可以分别建模为 d-q 电流的函数，如方程式 44 和方程式 45 所示。

$$L_d = f_1(i_d, i_q) = f_1(i_d) \tag{44}$$

$$L_q = f_2(i_q, i_d) = f_2(i_q) \tag{45}$$

通过简化方程式 41 来减轻 ISR 计算负担，基于电机参数的常数 K_{mtpa} 改为用方程式 47 表示，其中 K_{mtpa} 在后台循环中使用更新的 L_d 和 L_q 进行计算。

$$K_{mtpa} = \frac{\psi_m}{4 \times (L_q - L_d)} = 0.25 \times \frac{\psi_m}{(L_q - L_d)} \tag{46}$$

$$\beta_{mtpa} = \cos^{-1}\left(K_{mtpa}/I_s - \sqrt{(K_{mtpa}/I_s)^2 + 0.5} \right) \tag{47}$$

第二个中间变量 G_{mtpa} 由方程式 48 进行表示，用于进一步简化计算。使用 G_{mtpa} ，MTPA 控制的角度 β_{mtpa} 可以通过方程式 49 进行计算。这两个计算在 ISR 中执行，以获得真实的电流角度 β_{mtpa} 。

$$G_{mtpa} = K_{mtpa}/I_s \quad (48)$$

$$\beta_{mtpa} = \cos^{-1}\left(G_{mtpa} - \sqrt{G_{mtpa}^2 + 0.5}\right) \quad (49)$$

在所有情况下，都可以通过作用于直轴电流 i_d 来减弱磁通量以扩大可达到的转速范围。作为进入该恒定功率工作区域的结果，选择弱磁控制而不是在恒定功率和电压区域中使用的 MTPA 控制。由于最大逆变器电压受到限制，PMSM 电机无法在反电动势（几乎与永磁场和电机转速成正比）高于逆变器最大输出电压的转速区域中运行。在 PM 电机中，无法直接控制磁通量。不过，通过添加负 i_d 来减弱磁通量以扩大可达到的转速范围。考虑到电压和电流约束，电枢电流和端子电压会受到限制，如方程式 32 和方程式 33 所示。逆变器输入电压（直流链路电压）的变化限制了电机的最大输出。此外，最大基波电机电压还取决于所使用的 PWM 方法。在方程式 35 中，IPMSM 有两个因素：一个是永磁值，另一个是电感和磁通电流。

图 2-26 展示了用于实现弱磁的典型控制结构。β_{fw} 是弱磁 (FW) PI 控制器的输出，可生成基准 i_d 和 i_q。在电压幅度达到限制之前，FW 的 PI 控制器的输入始终为正，因此输出始终在 0 处达到饱和。

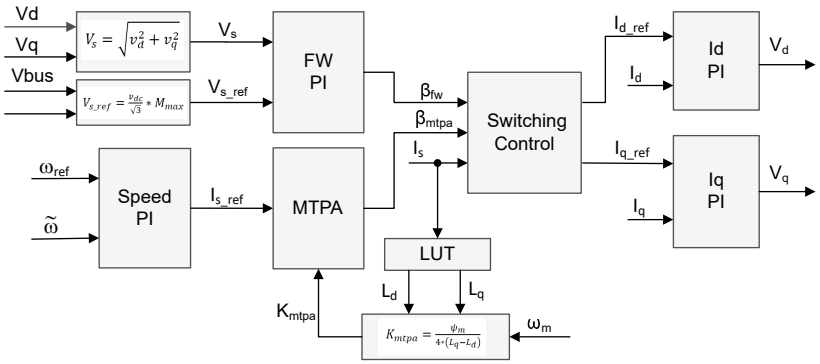


图 2-26. 弱磁和每安培最大扭矩控制的方框图

图 2-15 和图 2-17 展示了基于 FAST 或 eSMO 的 FOC 实现的方框图。这些方框图概述了 FOC 系统的功能和变量。电机驱动 FOC 系统中有两个控制模块：一个是 MTPA 控制，一个是弱磁控制。这两个模块根据输入参数分别生成电流角度 β_{mtpa} 和 β_{fw}，如图 2-27 所示。

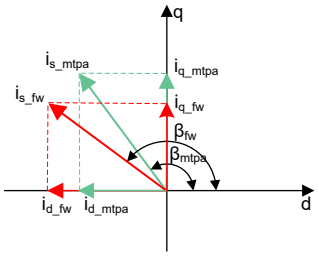


图 2-27. FW 和 MTPA 期间 IPMSM 的电流相量图

切换控制模块用于决定应用哪个角度，然后计算基准 i_d 和 i_q，如方程式 37 和方程式 38 所示。可以根据以下公式来选择电流角度：方程式 50 和方程式 51.

$$\beta = \beta_{fw} \, if \, \beta_{fw} > \beta_{mtpa} \quad (50)$$

$$\beta = \beta_{mpta} \, if \, \beta_{fw} < \beta_{mtpa} \quad (51)$$

图 2-28 是显示在主循环和中断中运行采用 FW 和 MPTA 的 InstaSPIN™-FOC 所需步骤的流程图。

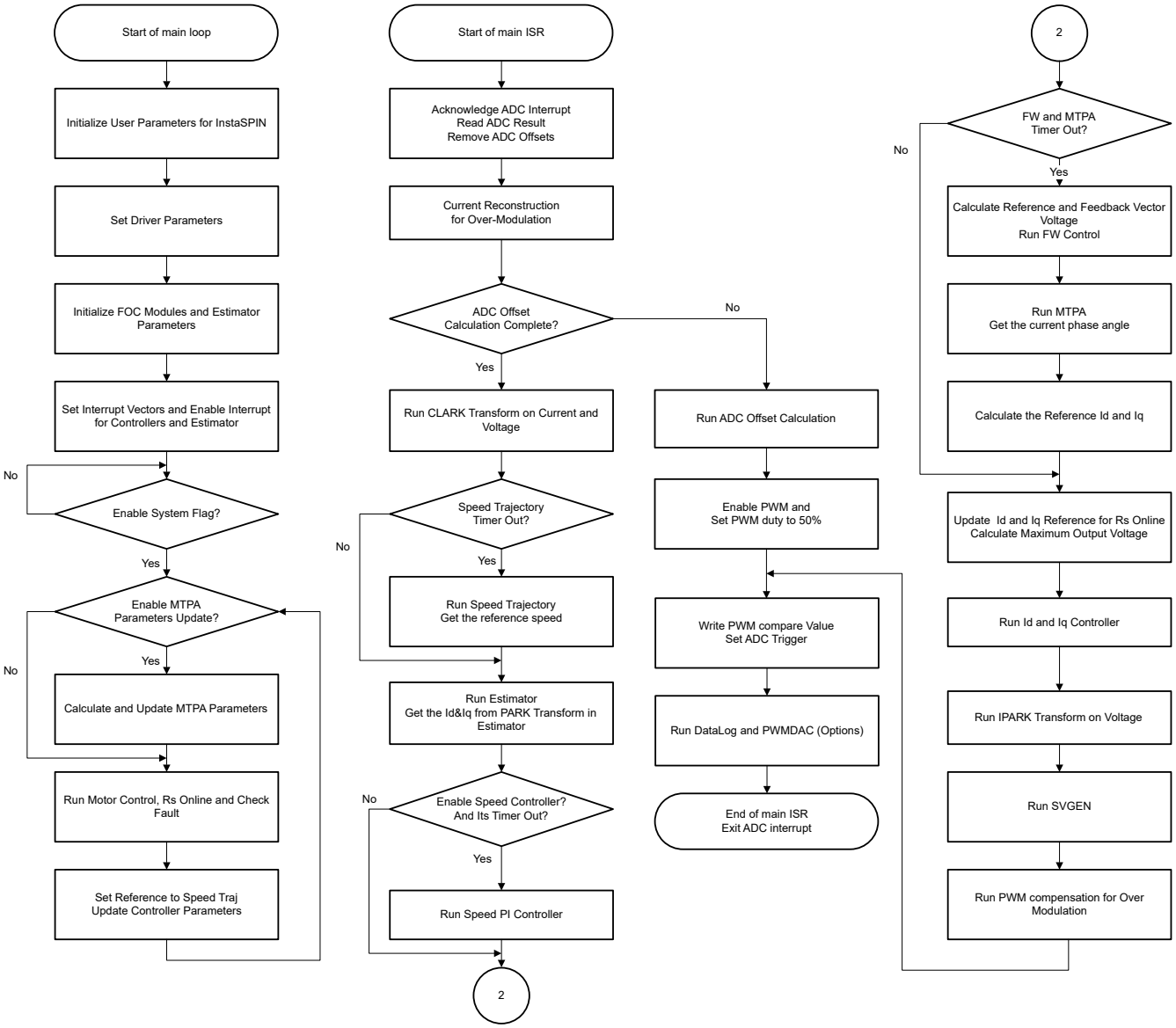


图 2-28. 采用 FW 和 MTPA 的 InstaSPIN-FOC 工程的流程图

2.4.2.4 电机驱动器的硬件必要条件

用于控制电机的算法利用电机条件的采样测量值，包括直流母线电源电压、每个电机相位上的电压、每个电机相位的电流。需要正确设置一些与硬件相关的参数，才能正确识别电机并使用磁场定向控制 (FOC) 有效地运行电机。以下各节说明如何计算采用 FAST 或 eSMO 的电机控制的电流标度值、电压标度值和电压滤波器极点。

2.4.2.4.1 电机电流反馈

支持两种技术来测量电机的相电流。

- 三分流器电流检测
- 单分流器电流检测

可以在工程的构建配置中选择这两种电流检测技术中的任何一种。*Flash_MtrInv_3SC* 构建配置支持三分流器电流检测方法，*Flash_MtrInv_1SC* 支持单分流器电流检测方法，如节 3.3.2 中所述。

2.4.2.4.1.1 三分流器电流检测

在每个 PWM 周期内，作为电机控制算法的一部分，微控制器会对流经电机的电流进行采样。TMS320F2800137 子板支持 1 至 3 个分流器电流检测，而 MSPM0G1507 子板支持 1 至 2 个分流器电流检测。为了测量电机相位的

双向电流（即正负电流），以下电路需要 1.65V 的基准电压。该失调基准电压由电压跟随器生成，如图 2-29 所示。

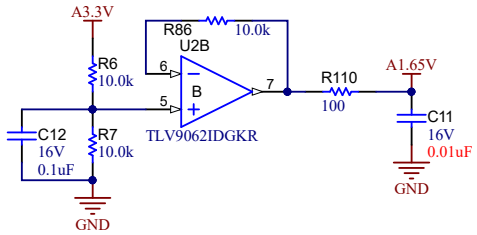


图 2-29. 3.3V 输入电路提供的 1.65V 基准

图 2-30 展示了电机电流如何表示为电压信号，其中包含滤波、放大和相对于 TMS320F2800137 子板 ADC 输入范围中心的偏移。该电路用于压缩机和风扇的三相 PMSM 的每一相。此电路的传输函数由 方程式 52 给定。

$$V_{OUT} = V_{OFFSET} + (I_{IN} \times R_{SHUNT} \times G_i) \tag{52}$$

其中

- R_{shunt} = 0.05Ω
- V_{offset} = 1.65V

利用计算出的电阻值，可得到图 2-36 所示的检测电路，G_i 由方程式 53 给出。

$$G_i = \frac{R_{fb}}{R_{in}} = \frac{R18}{(R97 + R15)} = \frac{10\ k\Omega}{20 + 2.4\ k\Omega} = 4.132 \tag{53}$$

微控制器可测量的最大峰峰值电流由方程式 54 给出。

$$I_{scale\_max} = \frac{V_{ADC\_max}}{R_{SHUNT} \times G_i} = \frac{3.3}{0.05 \times 4.132} = 15.97\ A \tag{54}$$

其峰峰值为 ±7.99A。以下代码片段显示了如何在 user_mtr1.h 文件中为压缩机电机定义该值：

```
//! \brief Defines the maximum current at the AD converter
#define USER_M1_ADC_FULL_SCALE_CURRENT_A         (15.97f)
```

正确的电流反馈极性也很重要，因为这样才能确保微处理器精确测量电流。在该硬件电路板配置中，分流电阻器的负引脚接地，同时与运算放大器的反相引脚连接。突出显示的符号需要在软件中配置为具有正确的电流反馈极性，如 user.mtr1.h 中的以下代码片段所示：

```
// define the sign of current feedback based on hardware board
#define USER_M1_SIGN_CURRENT_SF     (1.0f)
```

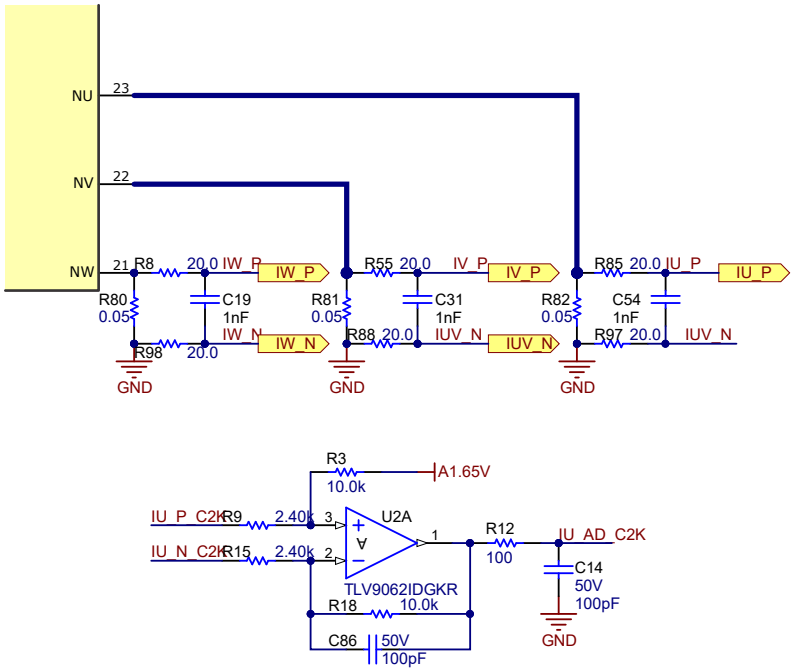


图 2-30. TMS320F2800137 的三分流器电流检测电路

MSPM0 子板上使用两个高端内部放大器实现了两个分流电流检测，以节省系统成本。放大器增益也为 4.132，截止频率为 70kHz。图 2-31 展示了 MSPM0G1507 子板的双分流器电流检测电路。

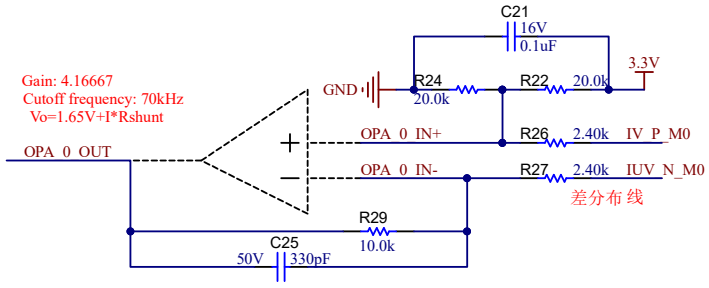


图 2-31. MSPM0G1507 双分流器电流检测电路

2.4.2.4.1.2 单分流器电流检测

单分流器电流检测技术测量直流链路总线电流，并在了解功率 FET 开关状态的情况下重建电机的三相电流。*使用单一直流链路分流器的 PMSM 无传感器 FOC* 应用手册中详细介绍了单分流器技术。

在该参考板上，通过移除两个分流器并短接电源模块的 U/V/W 接地连接来实现单分流器电流检测技术，如图 2-32 所示。

1. 在主板上，移除电流采样电阻 R81 和 R82，只保留电流采样电阻 R80 来检测直流链路电流。
2. 在 TMS320F2800137 子板上，移除 C86 以增加单分流器采样的 U2A 带宽。
3. 在 MSPM0G1507 子板上，移除 C29 以增加单分流器采样的带宽。
4. 使用粗导线将 NU、NV 和 NW 引脚连接在一起。

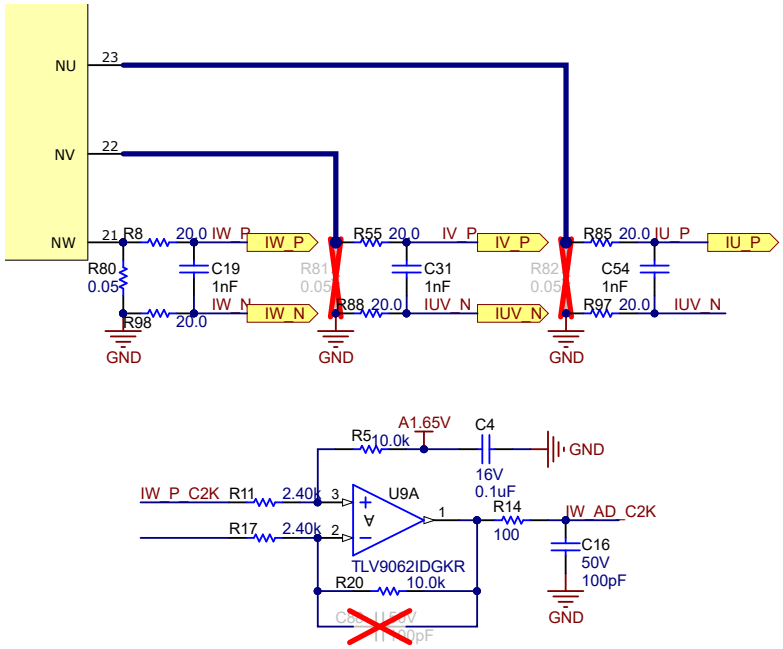


图 2-32. TMS320F2800137 的单分流器电流检测电路

图 2-33 展示了 MSPM0G1507 子板的单分流器电流检测电路。

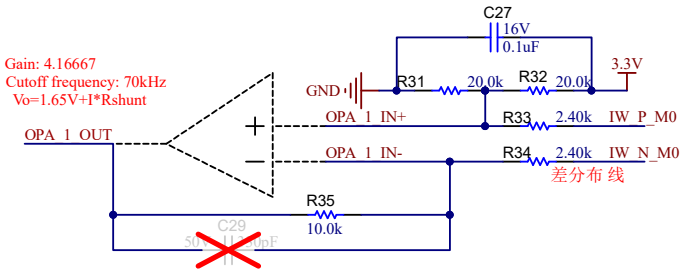


图 2-33. MSPM0G1507 单分流器电流检测电路

默认情况下，该电路板具有三个分流电阻器，图 2-34 展示了分流电阻器的布局。要使用单分流电阻器运行，请移除 R81 和 R82，同时保留 R80，将 NU、NV 和 NW（R80、R81 和 R82 的引脚 2）焊接在一起，则所有三相电流仅流经 R80。

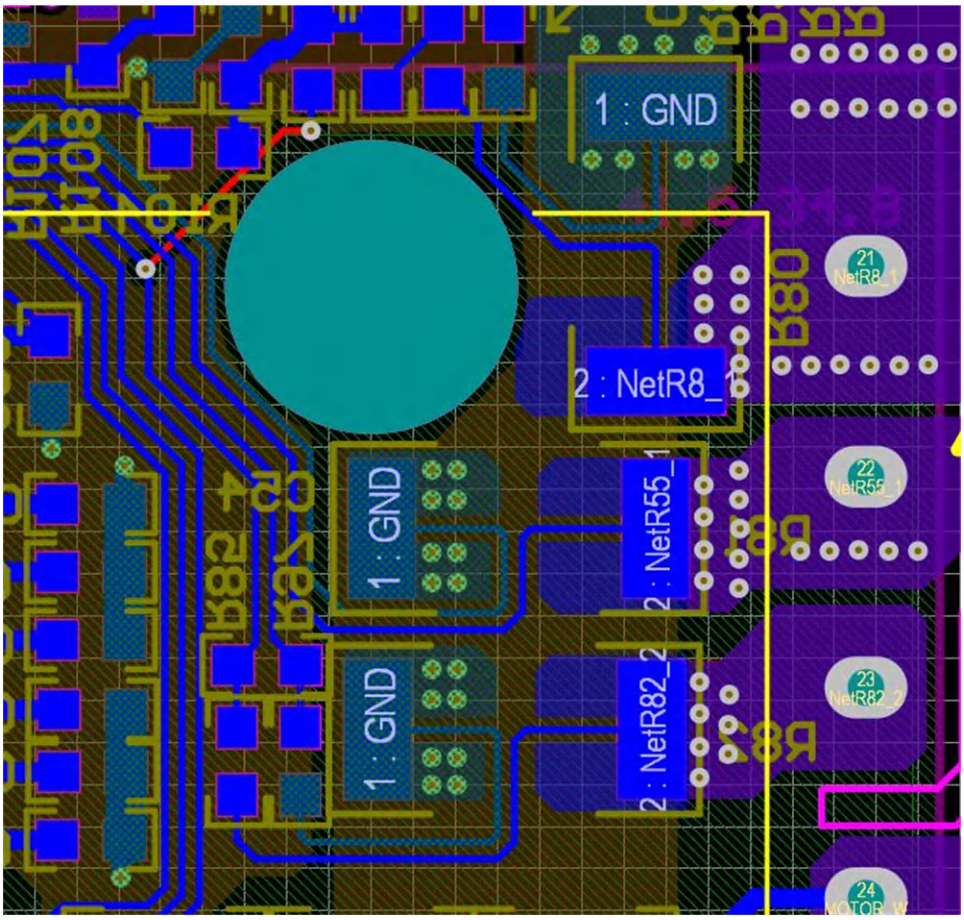


图 2-34. 分流电阻器布局

直流链路电流是单向信号，因此可以将直流电流偏移设置为最小值或最大值，以提高直流链路电流的 ADC 采样范围，如图 2-35 所示。在 TMS320F2800137 子板上，将 R7 从 10kΩ 更改为 1kΩ/1%，使基准电压具有 0.3V 失调电压，以进行直流电流检测。

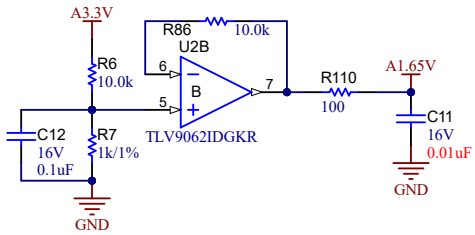


图 2-35. TMS320F2800137 子板单个分流器的直流偏移基准

该电流采样电路的传递函数和单分流器的计算与三分流器相同。

对于 MSPM0 子板，单分流器电流检测的偏移量也可以通过将 R31 从 20kΩ 降到 2kΩ 而降低到 0.3V，如图 2-33 所示。

2.4.2.4.2 电机电压反馈

FAST 估算器需要电压反馈，以在最宽的速度范围内实现最佳性能，相电压直接从电机相位测量，而不是使用软件估算。eSMO 依靠软件估算值来表示电压相位，而不使用电机相电压检测电路。此软件值 (USER_ADC_FULL_SCALE_VOLTAGE_V) 取决于感测电机相电压反馈的电路。图 2-36 展示了如何使用基于电阻分压器的电压反馈电路对电机电压进行滤波和调整以适应 ADC 输入范围。类似的电路用于测量全部三个压缩机和风扇电机以及直流母线。

考虑到 ADC 输入的最大电压为 3.3V，该参考设计中的微控制器可测量的最大相电压反馈可通过方程式 55 进行计算。

$$V_{FS} = V_{ADC\_FS} \times G_V = 3.3\ V \times 122.46 = 404.13\ V \quad (55)$$

其中

- G_V 是衰减因子，并且 G_V 可通过方程式 56 进行计算

$$G_V = \frac{(R62 + R67 + R70 + R74)}{R74} = \frac{(332\ k\Omega + 332\ k\Omega + 332\ k\Omega + 8.2\ k\Omega)}{8.2\ k\Omega} = 122.46 \quad (56)$$

对于该电压反馈电路，在 user_mtr1.h 中进行以下设置：

```
//! \brief Defines the maximum voltage at the AD converter
#define USER_M1_ADC_FULL_SCALE_VOLTAGE_V          (404.1292683f)
```

FAST 估算器中需要使用电压滤波器极点，以便准确检测电压反馈。使滤波器的电压足够低，以便能够滤除 PWM 信号，同时允许高速电压反馈信号通过滤波器。通常，使用几百 Hz 的截止频率便足以过滤掉 5kHz 至 20kHz 的 PWM 频率。只有在运行超高速电机时生成 kHz 量级相电压频率的情况下，才需更改硬件滤波器。

在该参考设计中，滤波器极点设置可以使用方程式 57 来计算：

$$f_{filter\_pole} = \frac{1}{(2 \times \pi \times R_{Parallel} \times C)} = 405.15\ Hz \quad (57)$$

Where,

$C = 47nF$

$$R_{Parallel} = \left(\frac{(332\ k\Omega + 332\ k\Omega + 332\ k\Omega) \times 8.2\ k\Omega}{(332\ k\Omega + 332\ k\Omega + 332\ k\Omega) + 8.2\ k\Omega}\right) = 8.133\ k\Omega$$

下面的代码示例显示了 user_mtr1.h 中是如何定义该极点的：

```
//! \brief Defines the analog voltage filter pole location, Hz
#define USER_M1_VOLTAGE_FILTER_POLE_Hz            (416.3602877f)
```

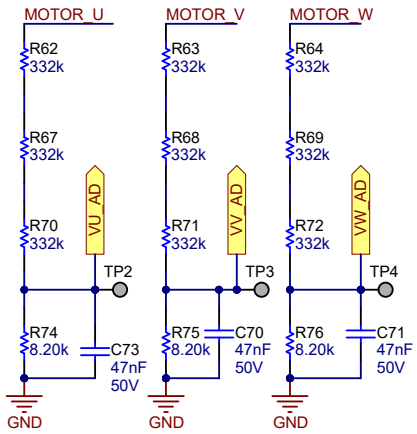


图 2-36. 电机电压检测电路

3 硬件、软件、测试要求和测试结果

3.1 入门硬件

本节详细介绍了设计电路板和软件测试和检验所需的设备、测试装置和过程说明。

3.1.1 硬件板概述

图 3-1 展示了典型电机逆变器系统的概述。

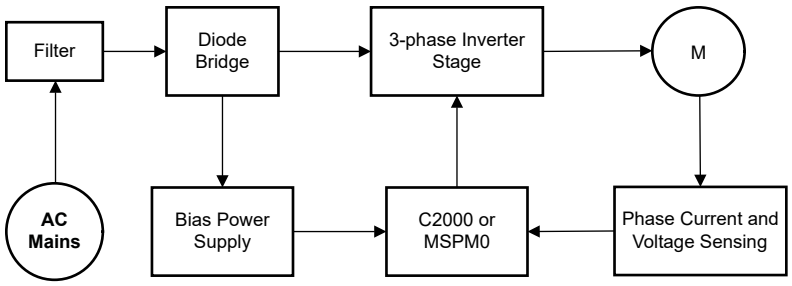


图 3-1. TIDA-010265 的硬件电路板方框图

电机控制板具有可实现完整电机驱动系统的功能组。以下是电路板上的块（及其功能）的列表，图 3-2 展示了电路板顶视图和 TIDA-010265 PCB 的不同块。

- 电源线输入滤波器
- 三相逆变器
  - 功率高达 750W 的三相逆变器支持 PMSM 或 IPM
  - 15kHz 开关频率
  - 1 至 3 个分流器电流检测
- 控件
  - 采用 48 引脚 LQFP 封装的单个 TMS320F2800137 或 MSPM0G1507 系列 MCU
  - 模拟信号的放大和输入滤波器
- 用于外部电机温度检测的接口
- 隔离式 UART 端口
- 辅助电源
  - 板载电源 +3.3V、+5V 和 +15V

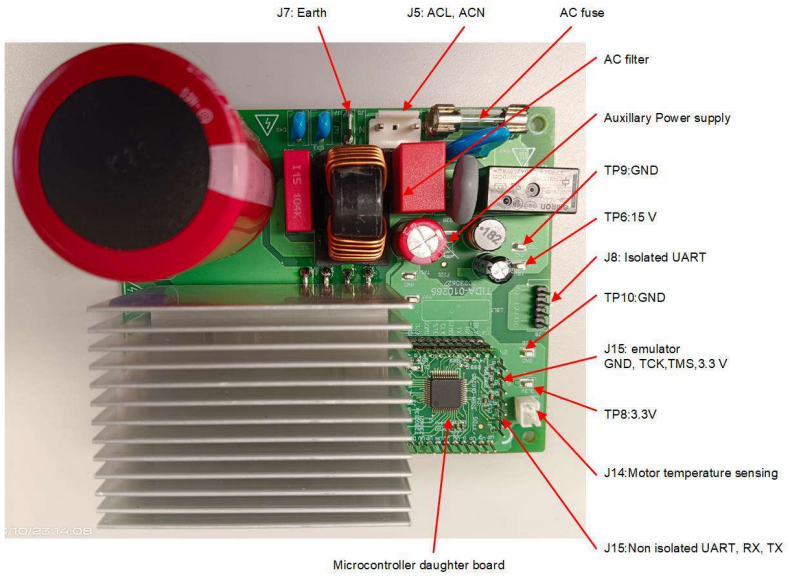


图 3-2. TIDA-010265 参考设计电路板布局布线

TI 建议在使用该板时采取以下预防措施：

警告

- 电路板通电时，请勿触摸电路板的任何部分或连接到电路板的元件。
- 使用交流电源或墙上电源为套件供电。TI 建议使用隔离交流电源。
- 通电时请勿触摸电路板、套件或组合件的任何部分。（尽管电源模块散热器与电路板隔离，但高电压开关会在散热器器身上产生一些电容耦合电压。）
- 控制接地可能很热。

3.1.2 测试条件

测试参考设计软件时请遵守以下规定：

- 对于输入，如果使用交流电源，则电源电压必须处于交流 165V 至 265V 范围内，如果使用直流电源，则电源电压必须处于 100V 至 400V 范围内。将输入交流电源的输入电流限制设置为 10A 或将直流电源设置为 6.5A，但在初始电路板启动期间先使用较低的电流限制。
- 对于输出，采用带测力计的三相 PMSM。

3.1.3 电路板验证所需的测试设备

设计人员必须使用以下设备进行电路板验证：

- 隔离式交流源
- 单相功率分析仪
- 数字示波器
- 万用表
- 直流电源
- 750W、三相 PM 同步电机
- 测力计
- 三相功率分析仪

3.2 入门 GUI

该参考设计提供了源代码，因此设计人员可以直接调试固件，如节 3.3 中所述。但是，软件调试需要更多时间。为了缩短开发时间，该参考设计提供了一个基于 UART 的 GUI 软件，以帮助快速调整任何定制应用的参数。本节介绍了如何使用 GUI 软件调试和调整电机控制参数。

目前，只有 C2000 子板固件支持 GUI。

通过 UART 将主机 PC 连接到该参考电路板时要小心，因为交流整流器会产生直流输出电压，该电压具有相对于保护性接地进行浮动的**热接地**。在将接地设备连接到套件时，必须使用隔离变压器。

3.2.1 测试设置

GUI 软件只需在主机 PC 和参考设计板之间进行 UART 连接。图 3-3 展示了用于使用 GUI 进行测试的硬件连接。按照以下步骤设置硬件：

1. 通过 UART 转 USB 适配器，仅将 J15 上的 TX、RX 和 GND 连接到主机 PC。适配器不需要提供 3.3V 电压。
2. 将电机线连接到 J10。
3. 连接万用表、示波器探头和其他测量设备，以探测或分析各种信号和参数
4. 通过将直流母线电源、交流电源或交流市电连接到 J5 和 J7 处的逆变器来为电路板供电。
   a. 直流电源的最大输出为 380V VDC。
   b. 交流电源的最大输出为 265V VAC（频率为 50/60Hz）。
   c. 交流市电为 220V VAC（频率为 50/60Hz）。

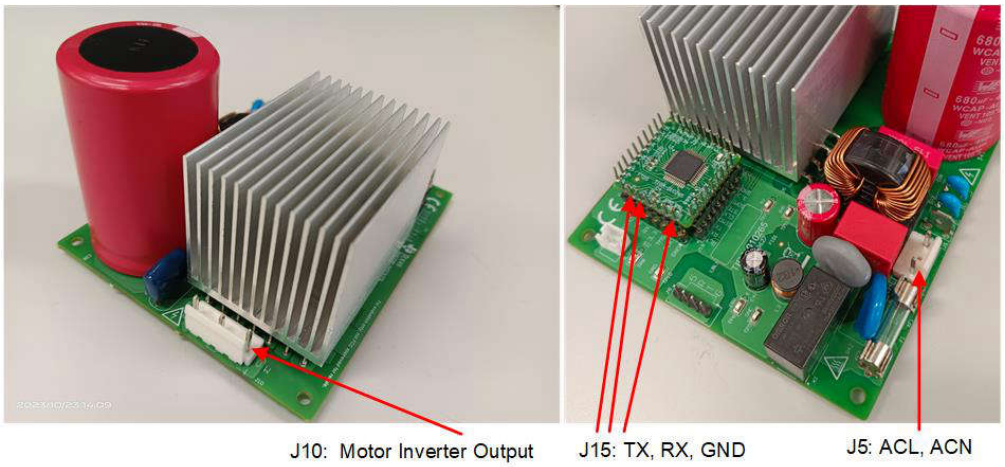


图 3-3. 使用 GUI 软件进行测试所需的硬件连接

备注

如果在测试时外部仿真器出现连接问题，请在 JTAG 信号和 USB 电缆上添加铁氧体磁珠。使连接线路尽可能短。

警告

两个电源域的接地平面可以相同或不同，具体取决于硬件配置。在将任何测试设备与电路板连接之前，应满足适当的隔离要求，以确保人身安全并防止损坏设备。在为电路板供电之前，请检查 GND 连接。如果测量设备连接至电路板，则需要隔离器。

3.2.2 GUI 软件概述

GUI 软件可以在基于 Microsoft® Windows® 的系统上运行。GUI 有 6 个选项卡：“Control Window”、“Debug Windows”、“Control Parameters”、“Motor Parameters”、“System Parameters”和“Communication Setting”，如图 3-4 所示。此版本的 GUI 中不提供“Analysis Windows”。这些选项卡提供多种功能，例如电机控制、识别、控制参数调优、虚拟示波器、读写 MCU 闪存等。

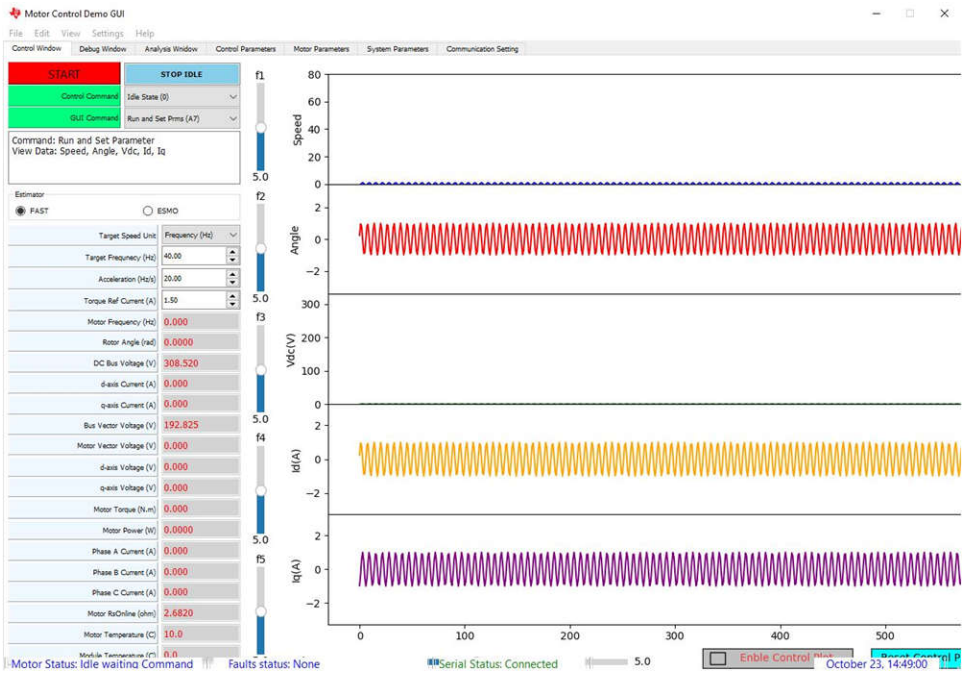


图 3-4. GUI 软件

3.2.3 设置串行端口

在主机 PC 上运行 GUI 软件，等待弹出 GUI 窗口。图 3-5 展示了通过 UART 将主机 PC 连接到该参考设计板的步骤。默认波特率为 258600bps。如果用户要使用不同的 UART 速度，则必须更改 GUI 和 C2000 上的波特率设置。

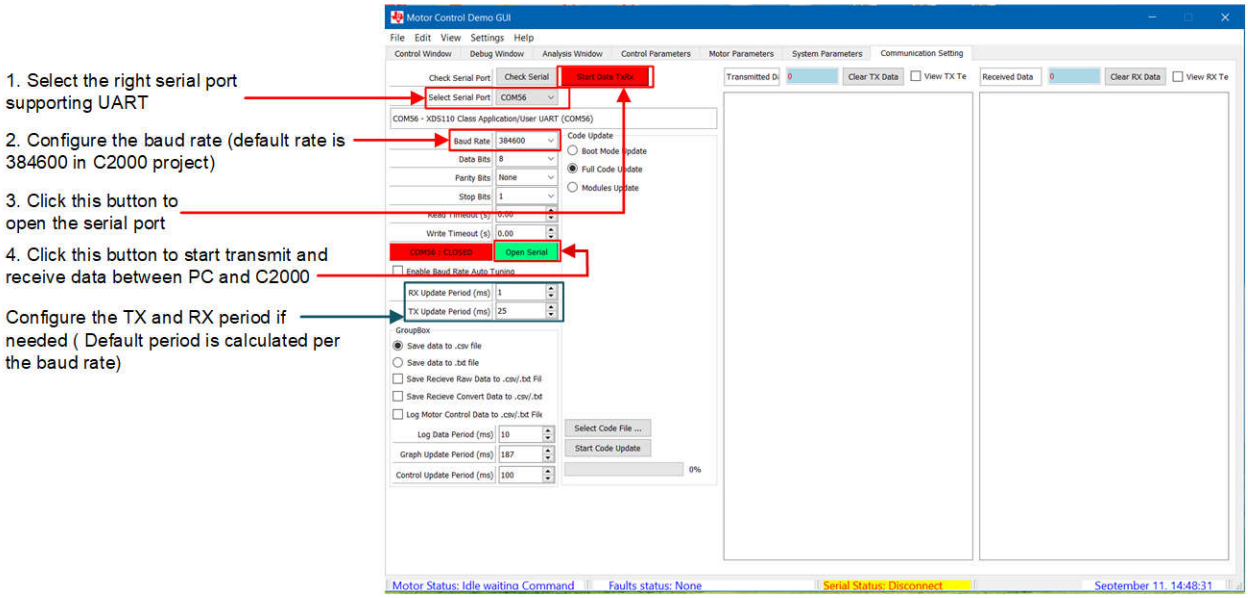


图 3-5. 设置串行端口

成功连接和通信可以通过 *Transmitted Data* 和 *Received Data* 的数量来验证。图 3-6 展示了连接成功后这两个数字都会持续增加。可以通过选中 *View Tx Text* 和 *View Rx Text* 来查看那些发送和接收的数据；但是，不要在电机运行时选中这些功能来查看。

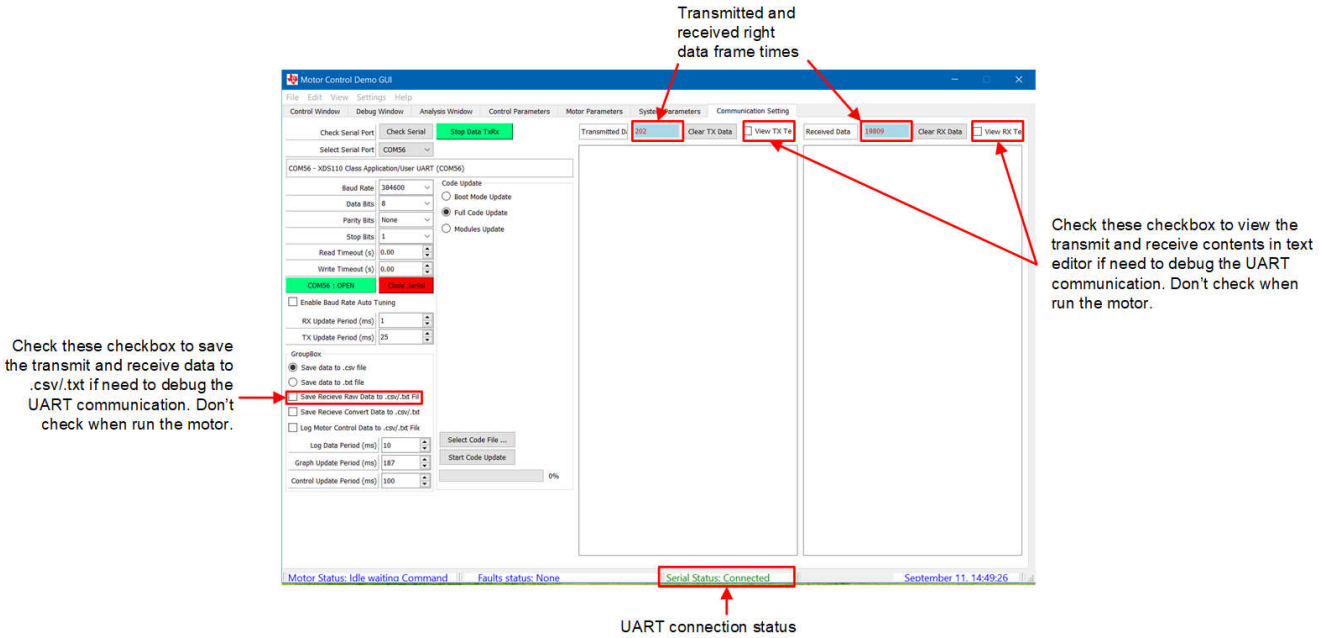


图 3-6. 成功通信状态

3.2.4 电机识别

采用正确的电机参数对于固件成功控制电机至关重要。这些参数包括定子电阻、定子电感、磁通等，并且这些参数对于默认电机在固件中具有默认值。

对于不同的 PMSM 电机，这些参数通常可以从规格中找到；但是，如果找不到这些参数，GUI 软件可以识别这些参数。

首先，在 *Control Window* 选项卡中选择电机识别命令，如图 3-7 所示。

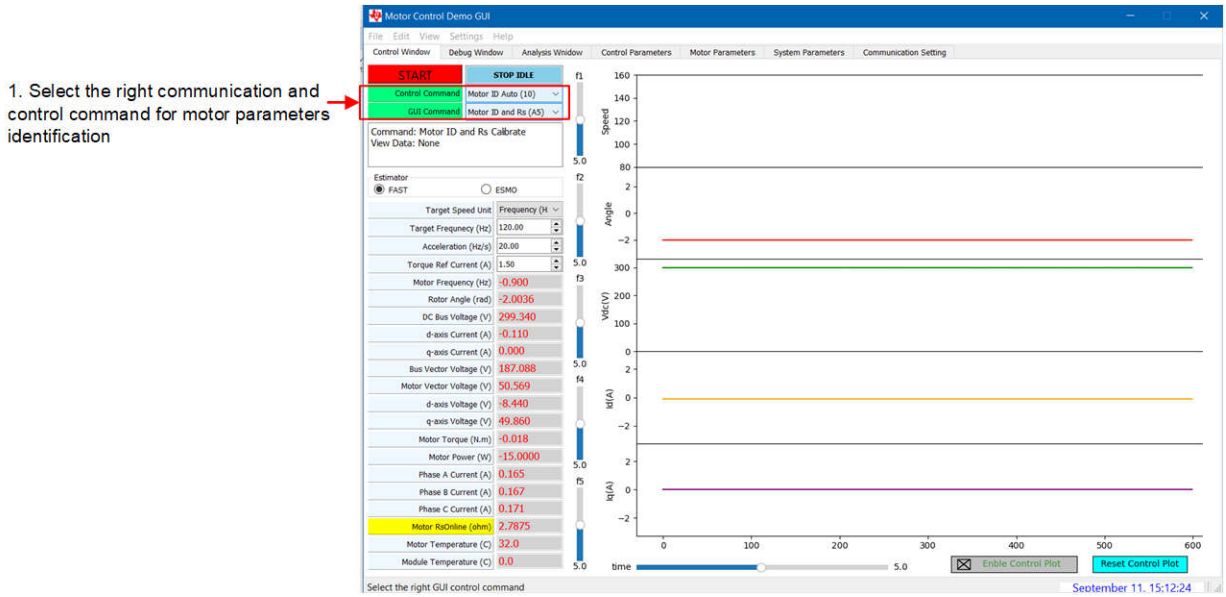


图 3-7. 电机识别命令

接下来，选择 *Motor Parameters* 选项卡。电机识别将电流施加到电机来估算电机参数，用户可以更改或保留这些识别参数，例如定子电阻估算电流、定子电感估算电流和 *R/L 激励频率 (Hz)*。

点击 *START* 按钮开始进行电机识别。此时会听到可闻噪声，并且电机会在识别期间低速运转。监控识别状态和电机参数，总识别时间约为 2 分钟。图 3-8 展示了启动电机识别的步骤。

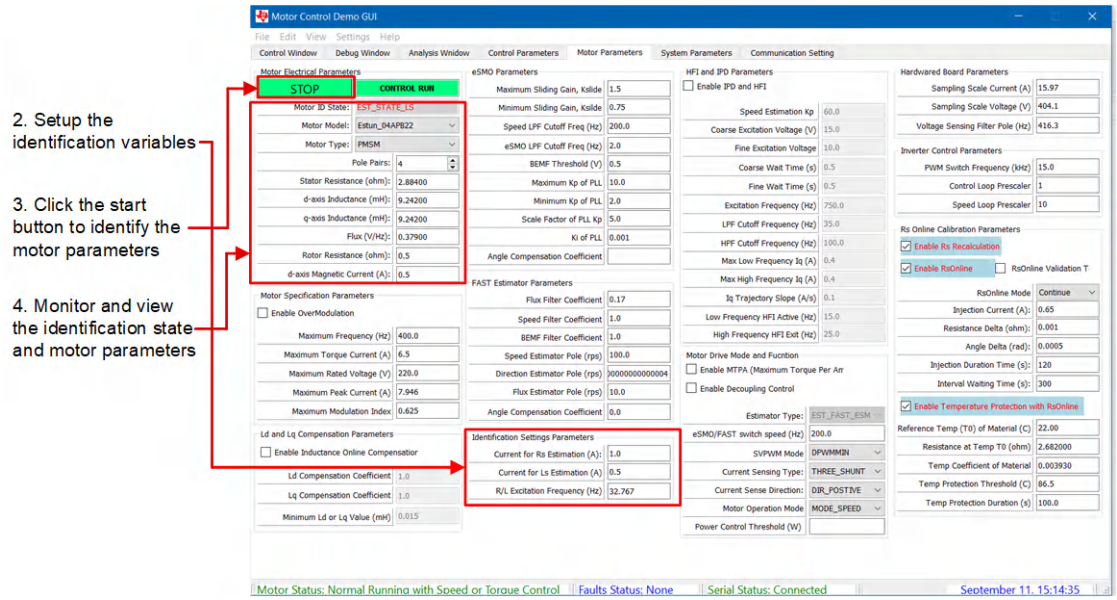


图 3-8. 启动电机识别

识别完成后，必须将电机对、定子电阻、定子电感、磁通等写入 MCU 闪存，以确保将这些参数保存在 MCU 中。选择 *Control Parameters* 选项卡，选择将电机和控制参数存储到 MCU 闪存中，或将其保存到文件中。要验证写入是否成功，请点击 *Read Settings from MCU Flash* 按钮，然后选择 *Motor Parameters* 选项卡，以确保电机参数与之前写入的参数相同。图 3-9 展示了本段中提到的按钮的位置。

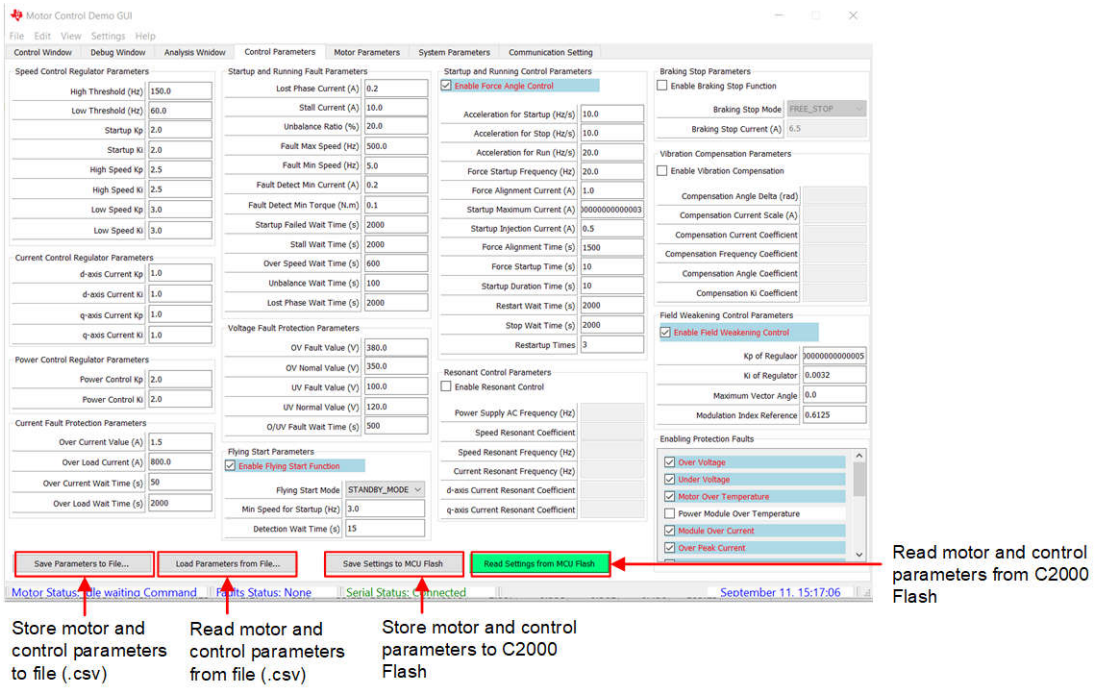


图 3-9. 存储电机识别结果

3.2.5 旋转电机

图 3-10 展示了 *Motor Parameters* 选项卡，其中指示了电机和控制参数。确保电机电气参数正确无误。

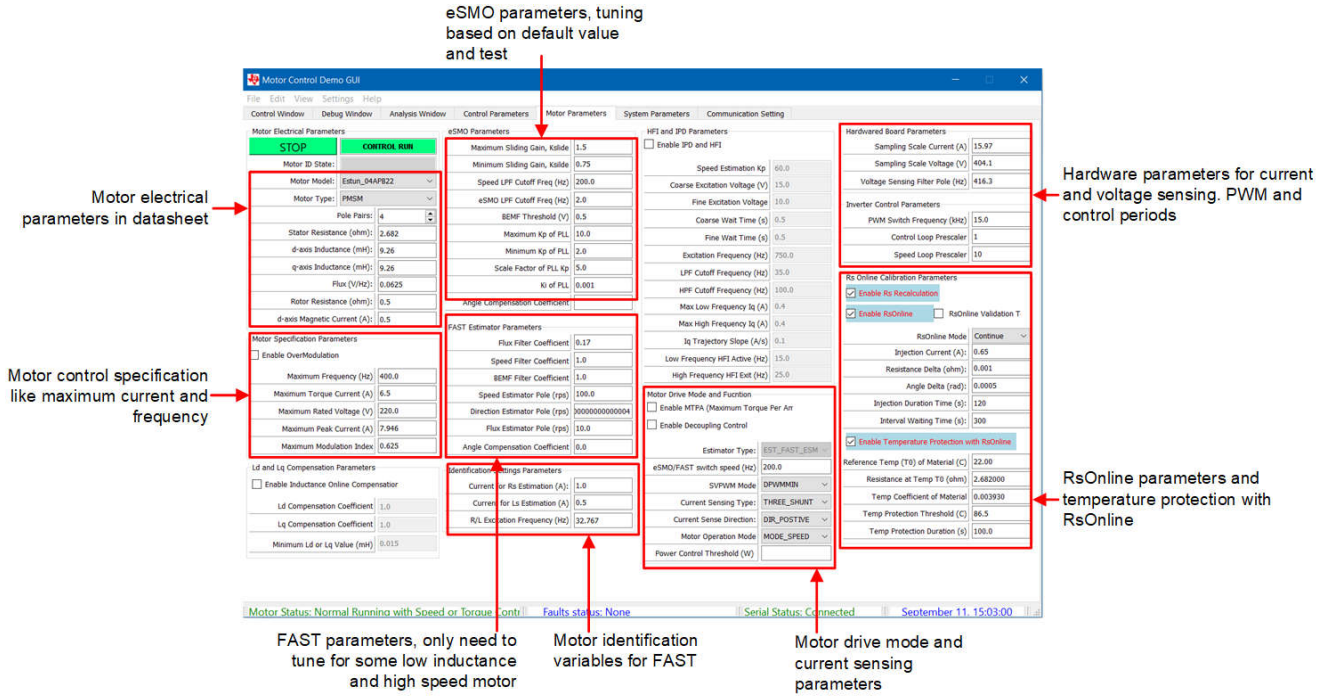


图 3-10. 电机驱动参数

在 *Control Window* 选项卡上，如果直流总线电压足够高 (>230 VDC)，*Control Command* 命令和 *GUI Command* 按钮为绿色，电机旋转命令有效。按照图 3-11 中的步骤旋转电机。

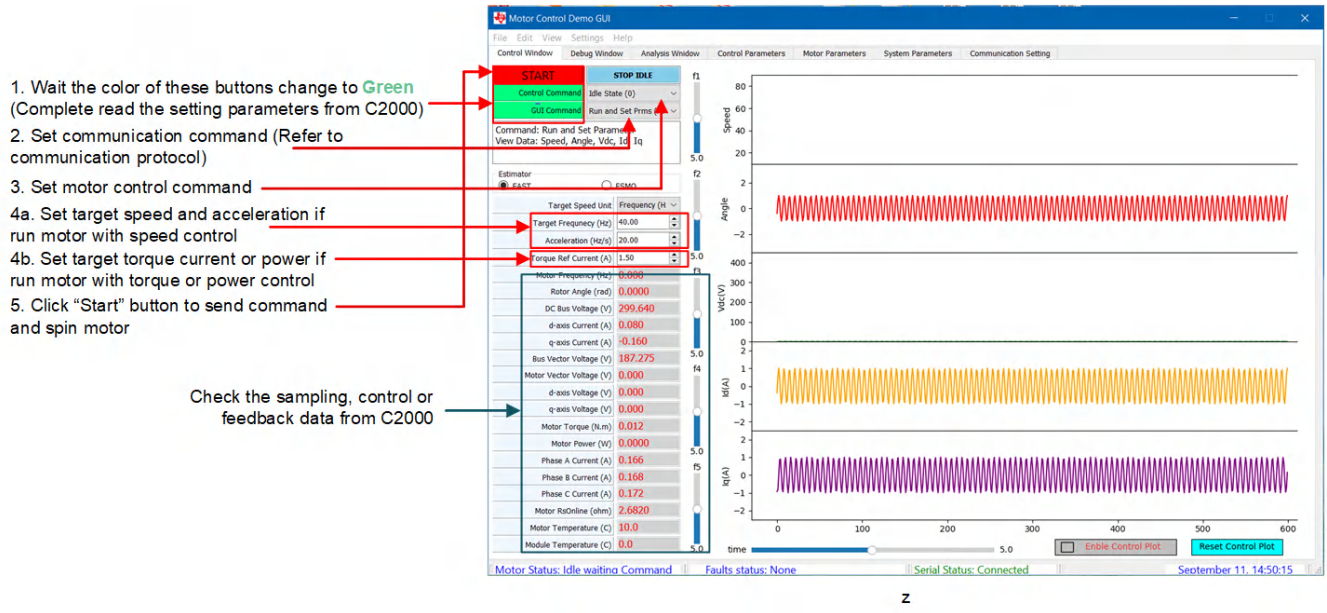


图 3-11. 旋转电机的步骤

3.2.6 电机故障状态

在电机旋转时可能会发现故障，尤其是当电机参数不正确或电机控制参数调整不当时。仔细观察 *Control Window*，并注意如图 3-12 所示报告的任何故障。

图 3-12. 故障状态监视

可以启用或禁用故障检测，如图 3-13 所示。

> **小心**
> 禁用某些故障可能会导致电路板损坏。在未充分了解可接受设置的情况下，请勿禁用故障保护。

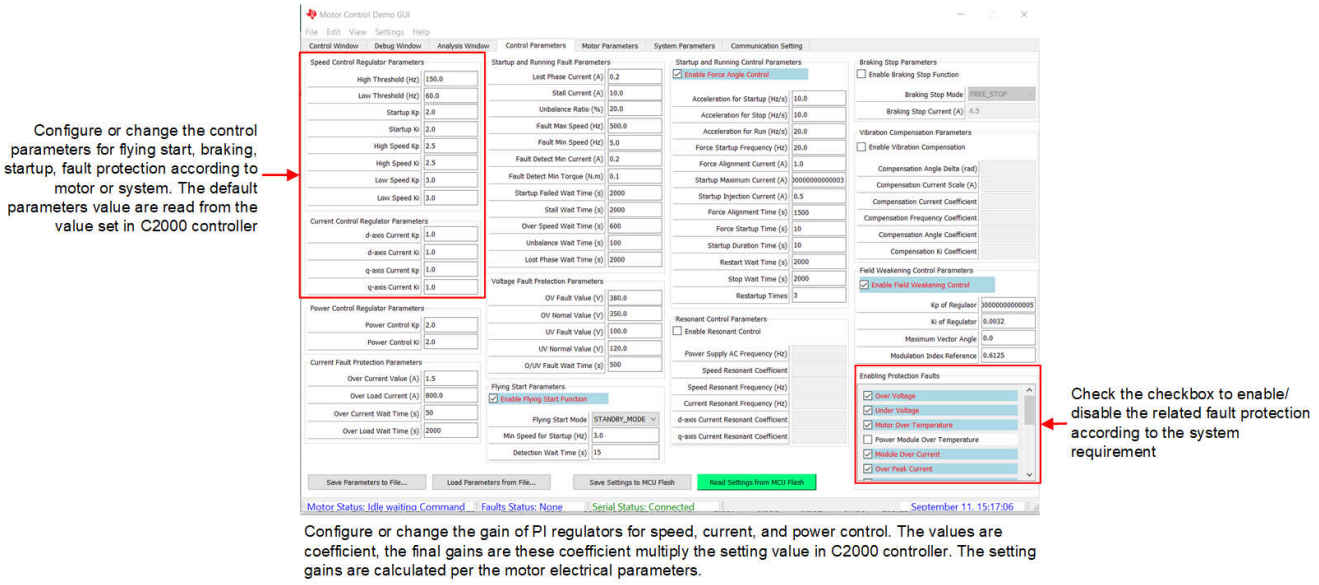


图 3-13. 电机控制参数和故障检测禁用

3.2.7 调整控制参数

在 *Debug Window* 选项卡上，可以调整 PI 稳压器的速度、电流和功率控制，如图 3-14 中所示。这些值是系数，最终的增益是这些系数乘以 C2000 控制器中的设定值。设置增益根据电机电气参数进行计算。如图 3-9 所示，这些调优值可以写入 MCU 闪存。

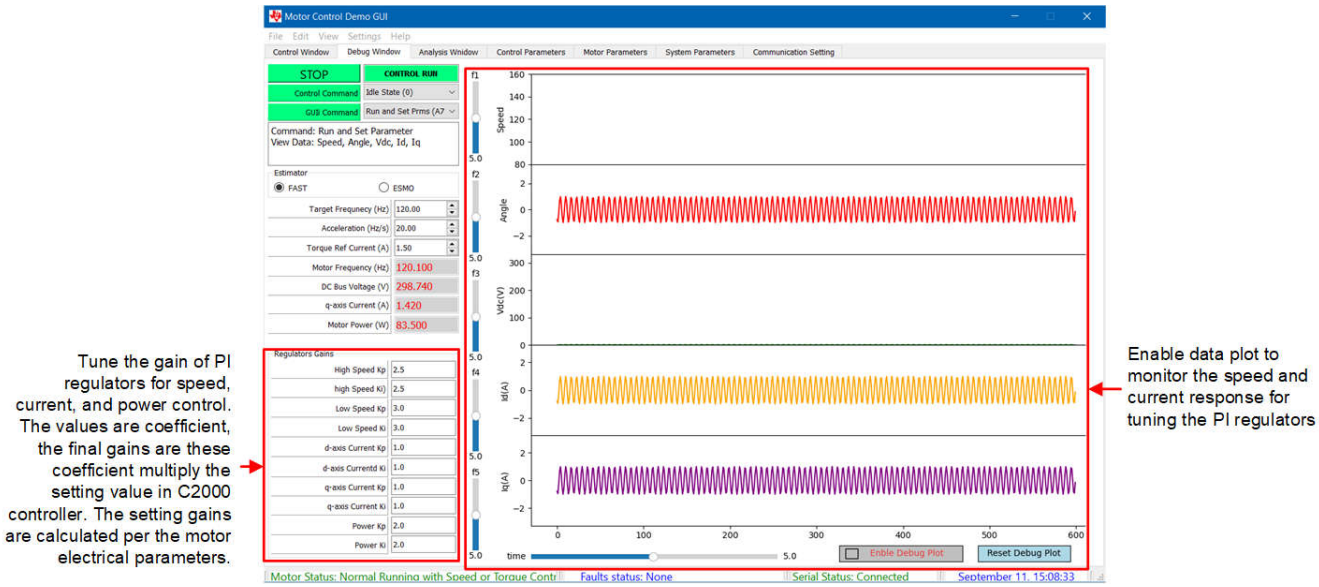


图 3-14. 调整控制参数

3.2.8 虚拟示波器

GUI 软件具有虚拟示波器功能，可以显示角度、相电流和相电压的波形。图 3-15 展示了如何配置命令以显示转子角度和相电流。

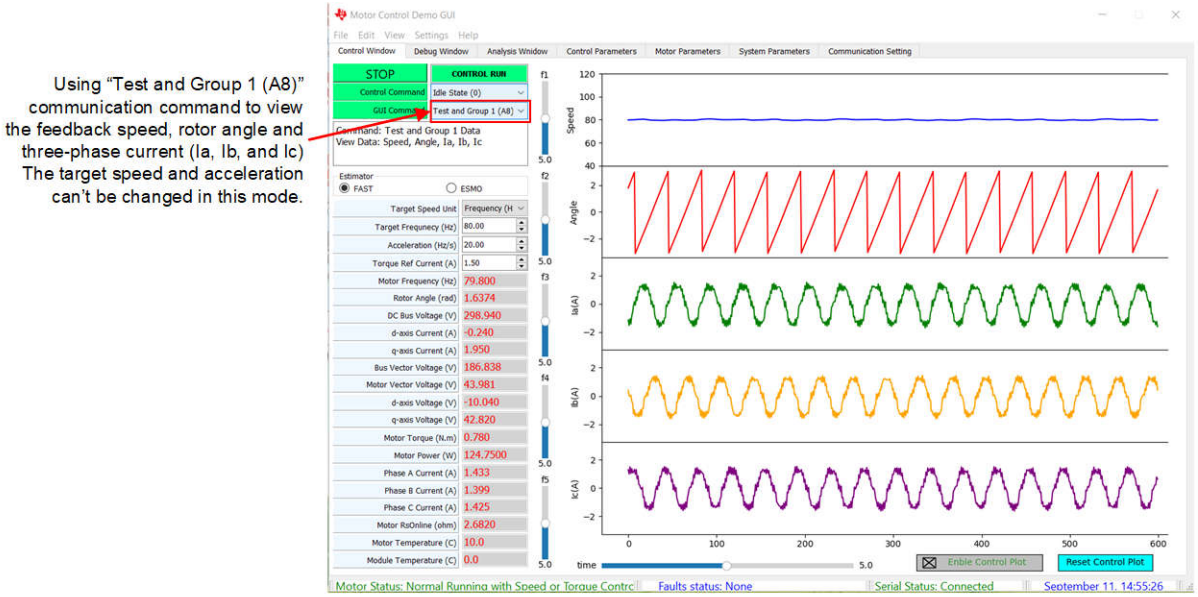


图 3-15. 用于转子角度和相电流的虚拟示波器

图 3-16 展示了命令 *Test and Group 2 (A9)* 并展示了转子角度和相位电压。

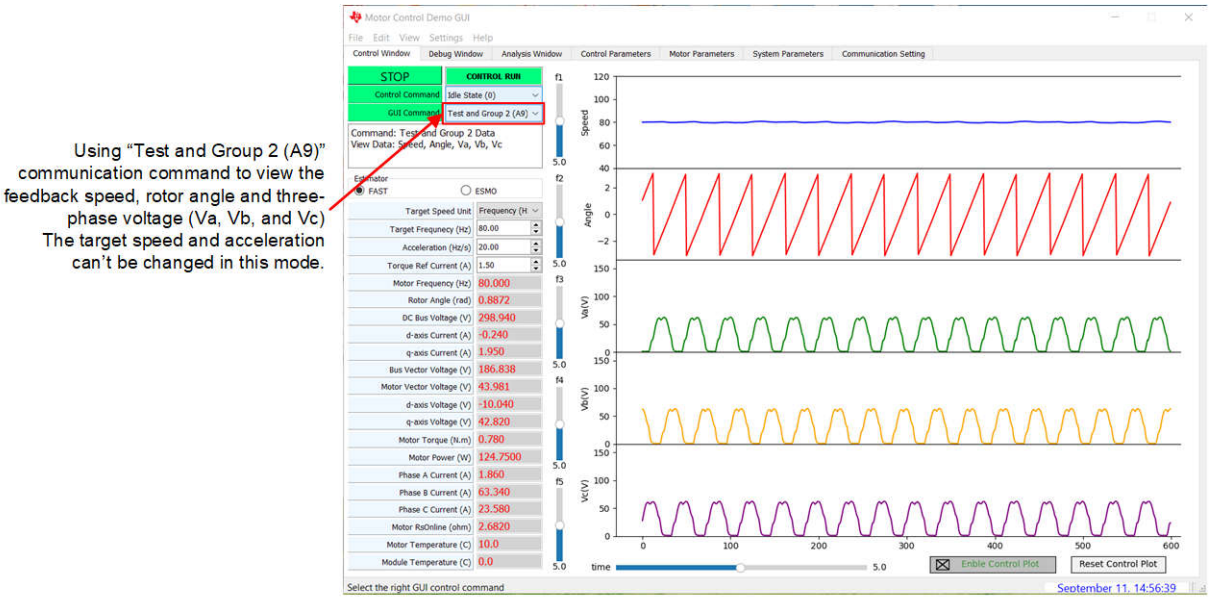


图 3-16. 用于转子角度和相电压的虚拟示波器

命令 *Test and Group 5 (AC)* 可以显示 FAST 和 EMO 的转子角度，如图 3-17 中所示。

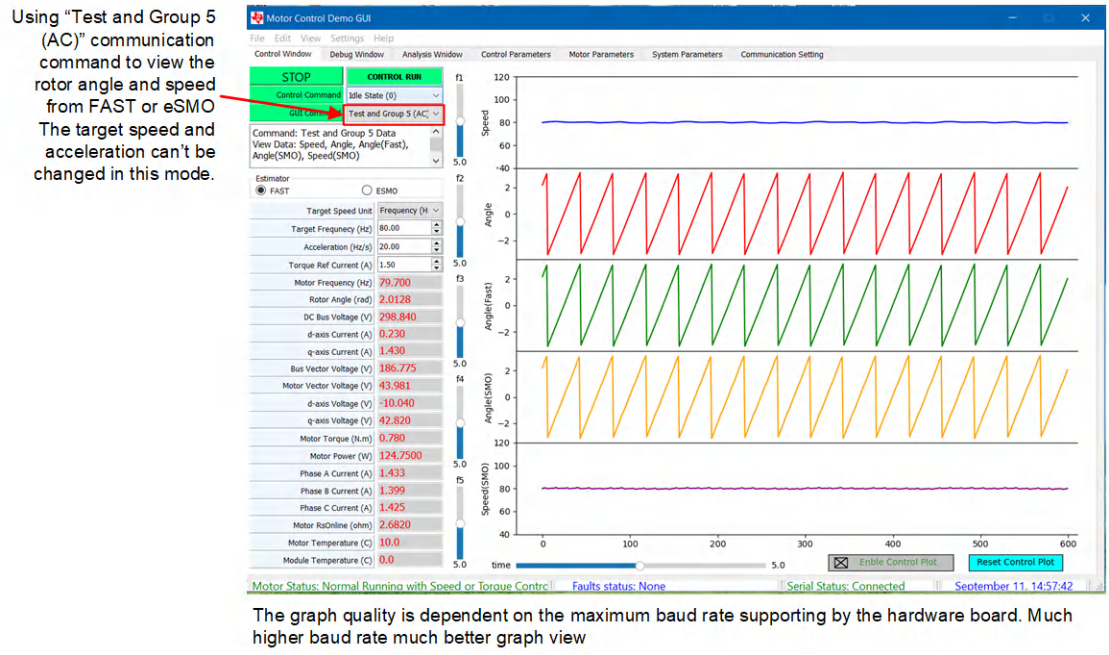


图 3-17. 用于 FAST 和 eSMO 的转子角度的虚拟示波器

3.3 C2000 固件入门

通过 TI 提供的链接下载 C2000WARE-MOTORCONTROL-SDK v5.01.00.00 或更高版本的软件，并进行安装。将该 Motor Control SDK 软件安装在默认文件夹中。该软件工程将保留在 C2000Ware Motor Control SDK 文件夹 `<install_location>\solutions\tida_010265_wminv\` 中。按照以下步骤使用不同的增量构建来构建和运行该代码。

3.3.1 下载并安装电路板测试所需的软件

1. 从 Code Composer Studio (CCS) 集成开发环境 (IDE) 工具文件夹下载 Code Composer Studio™ IDE 并进行安装。建议使用版本 12.5 或更高版本。
2. 通过以下两种方式之一安装 C2000WARE-MOTORCONTROL-SDK：
   - 通过 C2000Ware MotorControl SDK 工具文件夹下载软件
   - 转至 CCS 的 *View* → *Resource Explorer* 下。在 TI Resource Explorer 下，转至 *Software* → *C2000Ware_MotorControl_SDK*，然后点击安装按钮。
3. 安装完成后，关闭 CCS 并创建一个新的工作区以导入工程。

备注

该参考设计支持 SysConfig 配置器件引脚并在易于使用的图形界面中初始化器件外设。该功能仅供当前版本参考。设计人员可以下载 SysConfig，并参阅 C2000 SysConfig 软件指南以实施 SysConfig，从而将参考设计迁移至自己的板以配置器件。

3.3.2 在 CCS 内打开工程

基于 F280013x 的参考设计的 projectspec 文件位于以下目录中

```
<install_location>\solutions\tida_010265_wminv\f280013x\ccs\motor_control
```

在 CCS 中导入工程并通过右键点击工程名称来选择合适的构建配置，如图 3-18 所示。为 HVAC 参考设计选择合适的构建配置。*Flash_MtrInv_3SC* 构建配置支持三分流器电流检测方法，*Flash_MtrInv_1SC* 支持单分流器电流检测方法。

通过点击 *Project* → *Import CCS Projects*，配置工程以选择工程中的支持功能，然后对于基于 F280013x 的参考设计，浏览到 `<install_location>\solutions\tida_010265_wminv\f280013x\ccs\motor_control` ，接着右键点击导入的工程名称，再点击 *Properties* 命令以为工程设置预定义的符号，如图 3-19 所示。

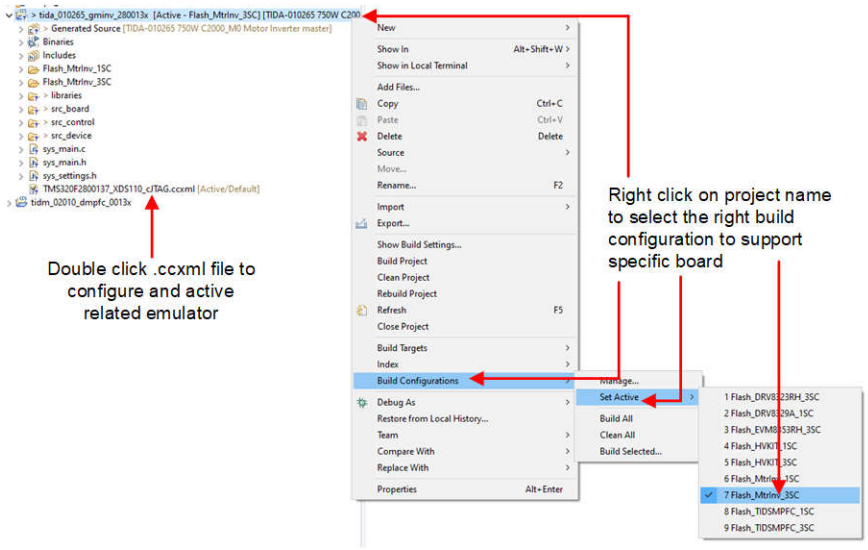


图 3-18. 选择合适的构建配置

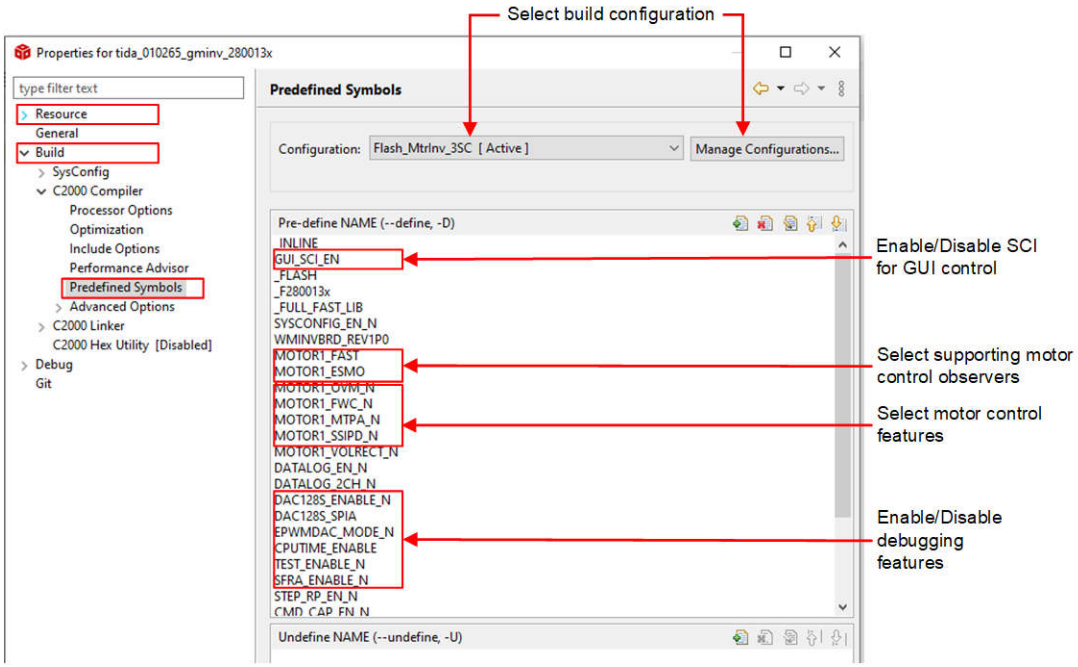


图 3-19. 在工程属性中选择合适的预定义符号

3.3.3 工程结构

导入工程后，CCS 内将显示 Project Explorer，如图 3-20 所示。器件外设配置基于 C2000Ware driverlib。用户只需更改 *hal.c* 和 *hal.h* 中的代码和定义。

文件夹 *src_control* 包含 *hal.c* 和 *user_mtr1.c*，用户可以在其中更改代码和定义。

文件夹 *src_board* 包含该硬件板的板驱动程序。

src_control 文件夹包含电机驱动文件，这些文件在中断服务例程和后台任务中调用电机控制核心算法函数。

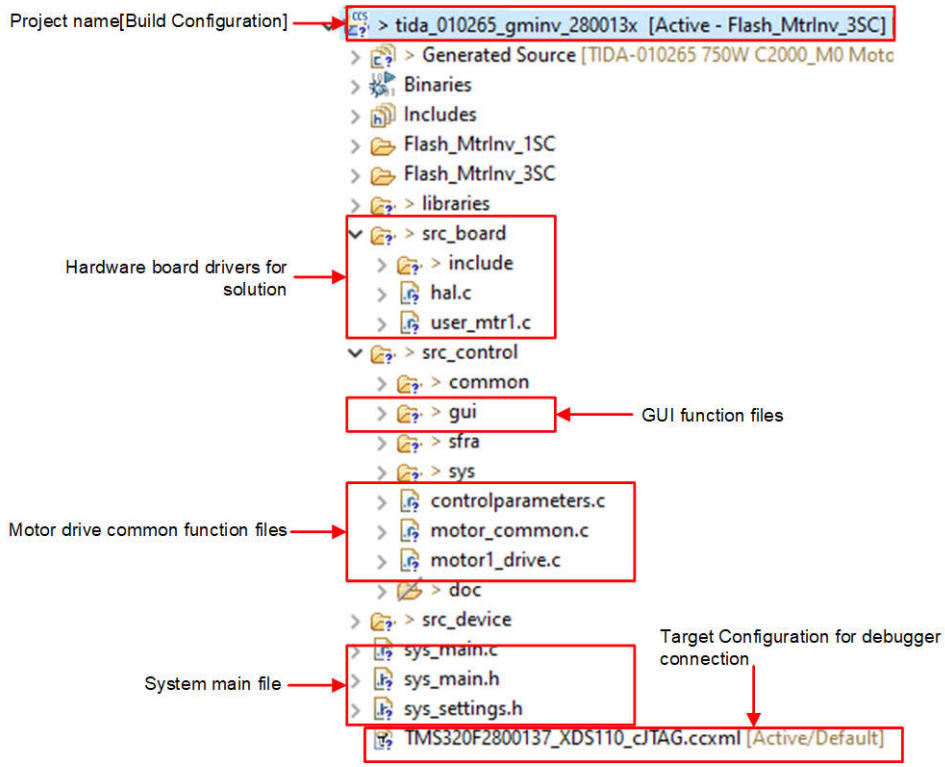


图 3-20. TIDA-010265“Project Explorer”视图

图 3-21 展示了工程软件流程图，其中包括用于电机控制的 ISR、一个主循环用于在后台循环中更新电机控制参数。

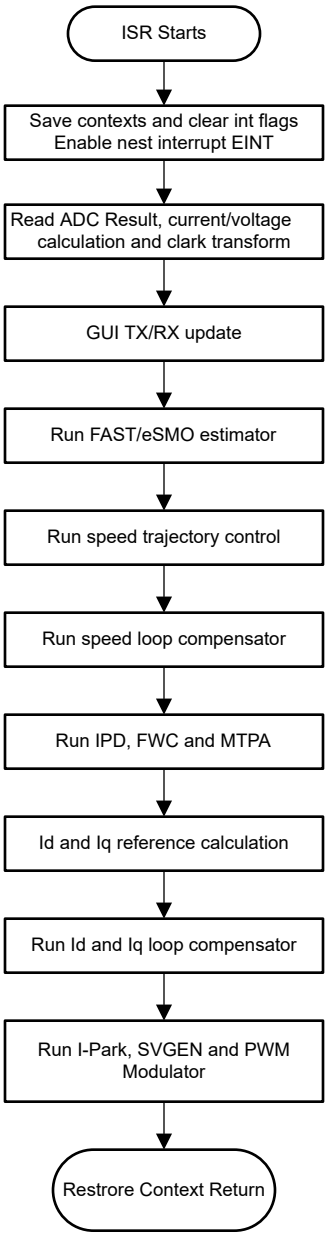


图 3-21. 固件工程流程图

该工程包含一个电机控制中断服务例程，每个 PWM 周期都会调用该例程。一些后台任务以一直循环的方式在 main() 进行调用，可用于运行不要求绝对计时精度的缓慢任务（如电机控制参数更新等）。CPU 计时器用于触发慢速后台任务。

motor1CtrlISR 被保留，用于调用电机驱动控制算法以旋转电机 1，该电机以 USER_M1_ISR_FREQ_Hz 的频率定期触发。

为了简化系统开发和设计，该参考设计的软件在四个实验室中通过增量构建 (DMC_BUILDLEVEL) 进行了组织，这使得学习和熟悉电路板和软件变得更加容易。这个方法对也适用于调试和测试电路板。表 3-1 列出了详细的增量构建选项。要选择特定的构建选项，请在 *sys_settings.h* 中选择相应的 BUILDLEVEL 选项。选择构建选项后，通过选择 ***rebuild all*** 编译器选项来编译工程。节 3.3.4 提供了有关运行每个构建级别选项的更多详细信息。

表 3-1. 递增构建选项

操作	构建选项	说明
电机驱动	DMC_LEVEL_1	50% PWM 占空比，验证 ADC 失调电压校准、PWM 输出和相移
	DMC_LEVEL_2	开环 v/f 控制，用于检查电机的电流和电压检测信号
	DMC_LEVEL_3	闭合电流环路，用于检查硬件设置
	DMC_LEVEL_4	电机参数识别，使用 InstaSPIN-FOC 或 eSMO 运行

3.3.4 测试步骤

警告

电路板上存在**高压**。要安全地评估该板，请使用适当的隔离和限流电源。在向板施加电源之前，必须在输出端连接适当的阻性负载或电子负载。请勿在通电时操作设备。仅使用具有适当额定值的设备并遵循适当的隔离和安全措施。

小心

将示波器和其他测试设备连接到电路板时要小心，因为交流整流器会产生直流输出电压，该电压具有相对于保护性接地进行浮动的**热接地**。在将接地设备连接到套件时，必须使用隔离变压器。

3.3.4.1 构建级别 1：CPU 和电路板设置

了解该构建级别中的目标：

- 评估系统的开环运行
- 使用 HAL 对象设置 MCU 控制器并初始化逆变器
- 验证 PWM 和 ADC 驱动器模块
- 熟悉 CCS 的操作

由于该系统以开环控制方式运行，因此 ADC 测量值只用于该构建级别的仪器用途。该构建级别仅使用 MCU 控制器和栅极驱动器的偏置电源。逆变器不提供高电压交流和直流电源。

在该构建级别中，电路板以开环方式执行（采用固定占空比）。电机的占空比设置为 50%。该构建级别验证来自功率级的反馈值检测以及 PWM 栅极驱动器的运行，并确保没有硬件问题。此外，可以在该构建级别中执行输入和输出电压检测校准。图 3-22 展示了该构建级别的软件流程。

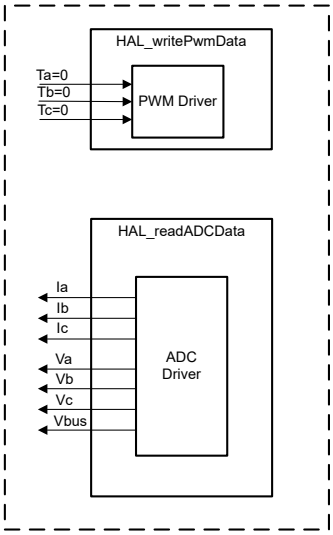


图 3-22. 控制软件方框图：构建级别 1 - 偏移验证

3.3.4.1.1 启动 CCS 并打开工程

要启动 CCS 并打开工程，请完成以下步骤：

1. 在 J15 处连接仿真器。
2. 将交流或直流电源连接到 J5，如图 3-23 中所示。
3. 打开 CCSv12.5（或更高版本）。工程包含开发可执行输出文件 (.out)（可在基于 C2000 控制器的硬件上运行）所需的所有文件和构建选项。在菜单栏中，点击 *Project* → *Import CCS Projects*。在 *Select search-directory:* 下，浏览至 C2000Ware Motor Control SDK 文件夹，然后选择 `<install_location>\solutions\tida_010265_wminv`。点击 *Finish* 来将相关工程导入 CCS。此工程调用所有需要的工具（编译器、汇编器、链接器）来构建工程。
4. 在左侧的工程窗口中，点击工程左侧的加号符号 (+)。图 3-20 展示了一个示例工程窗口。

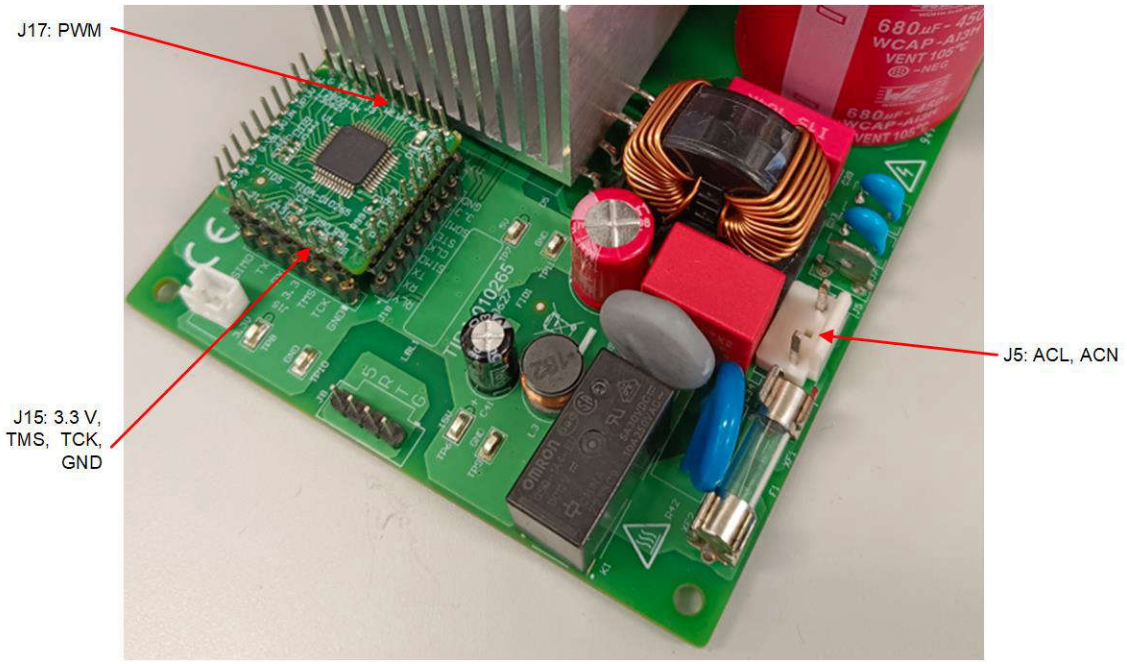


图 3-23. 连接外部交流或直流电源以验证硬件

3.3.4.1.2 构建和加载工程

要构建和加载工程，请完成以下步骤：

1. 右键点击工程名称，点击 *Properties* 命令，转到预定义的符号，以将 *GUI_SCI_EN* 更改为 *GUI_SCI_N*，从而为 GUI 禁用 SCI 功能，如图 3-19 所示。

   备注

   如节 3.2 中所述，可以再次启用 SCI 功能以允许 GUI 通过 SCI 进行控制。
2. 打开 sys_settings.h 文件，将 DMC_BUILDLEVEL 设置为 DMC_LEVEL_1。
3. 如果之前构建了另一个构建选项，则右键点击工程名称并点击 *Clean Project*，然后点击 *Build Project*。观察构建窗口中运行的工具。项目成功构建。
4. 在 *Project Explorer* 中，确保将正确的目标配置文件设置为“Active”，如图 3-20 所示。
5. 打开交流或直流电源，向 J5 施加 30VAC 或 40VDC，从而为控制器和栅极驱动器产生 +15V 和 3.3V。点击“Debug”按钮 或点击 *Run* → *Debug*。此时构建级别 1 代码可以编译并加载到 C2000 器件上。请注意右上角的“CCS Debug”图标，该图标表明用户现在处于 *Debug Perspective* 视图中。程序可以在 main() 的开始处停止。

3.3.4.1.3 设置调试环境窗口

在调试代码时观察本地和全局变量是一种标准的调试做法。在 CCS 中有多种不同的方法来实现这一做法，例如存储器视图和监视视图。此外，CCS 能够制作时域（和频域）图。该功能允许用户使用图形工具查看波形。

1. 点击菜单栏上的 *View* → *Expressions* 打开一个 *Expressions* 监视窗口。将鼠标移至 *Expressions* 窗口以查看工程中使用的变量。向 *Expressions* 窗口中添加变量，如图 3-24 中所示。该窗口使用在声明期间与变量关联

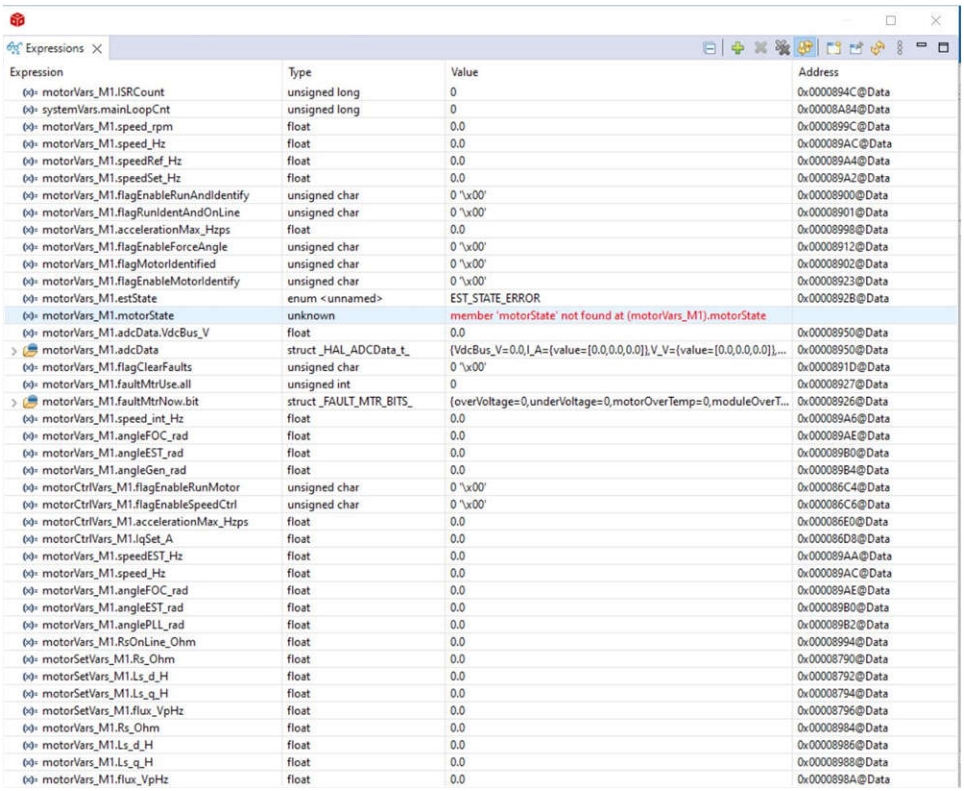
的数字格式，图 3-24 展示了一个 *Expressions* 窗口示例。通过右键点击表达式并进行选择来为变量选择所需的数字格式。

2. 或者，可以通过右键点击 *Expressions* 窗口并点击 *Import* 将一组变量导入到 *Expressions* 窗口中，然后浏览至工程目录 `<install_location>\solutions\tida_010265_wminv\src_control\common\debug`，选择 *BuildLevel1.txt* 并点击 *OK* 按钮以导入图 3-24 中所示的变量。

备注

此时主代码中的某些变量尚未初始化，可能包含一些无用的值。

3. 结构变量 *motorVars_M1[]* 引用了大多数与控制电机相关的变量。通过展开该变量，您可以查看所有变量并根据需要进行编辑。
4. 点击“Expressions”窗口中的 *Continuous Refresh* 按钮  。这将启用窗口的实时运行模式。通过点击该 *Expressions* 窗口中的下拉箭头，您可以选择 *Customize Continuous Refresh Interval* 并编辑该“Expressions”窗口的刷新率。选择过短的刷新间隔可能会影响性能。



Expressions

Expression	Type	Value	Address
motorVars_M1.ISRCount	unsigned long	0	0x0000894C@Data
systemVars.mainLoopCnt	unsigned long	0	0x00008A84@Data
motorVars_M1.speed_rpm	float	0.0	0x0000899C@Data
motorVars_M1.speed_Hz	float	0.0	0x000089AC@Data
motorVars_M1.speedRef_Hz	float	0.0	0x000089A4@Data
motorVars_M1.speedSet_Hz	float	0.0	0x000089A2@Data
motorVars_M1.flagEnableRunAndIdentify	unsigned char	0 '\x00'	0x00008900@Data
motorVars_M1.flagRunIdentAndOnLine	unsigned char	0 '\x00'	0x00008901@Data
motorVars_M1.accelerationMax_Hzps	float	0.0	0x00008998@Data
motorVars_M1.flagEnableForceAngle	unsigned char	0 '\x00'	0x00008912@Data
motorVars_M1.flagMotorIdentified	unsigned char	0 '\x00'	0x00008902@Data
motorVars_M1.flagEnableMotorIdentify	unsigned char	0 '\x00'	0x00008923@Data
motorVars_M1.estState	enum <unnamed>	EST_STATE_ERROR	0x0000892B@Data
motorVars_M1.motorState	unknown	member 'motorState' not found at (motorVars_M1).motorState	
motorVars_M1.adcData.VdcBus_V	float	0.0	0x00008950@Data
motorVars_M1.adcData	struct _HAL_ADCData_t_	{VdcBus_V=0.0,I_A={value=[0.0,0.0,0.0],V_V={value=[0.0,0.0,0.0],...	0x00008950@Data
motorVars_M1.flagClearFaults	unsigned char	0 '\x00'	0x0000891D@Data
motorVars_M1.faultMtrUse.all	unsigned int	0	0x00008927@Data
motorVars_M1.faultMtrNow.bit	struct _FAULT_MTR_BITS_	{overVoltage=0,underVoltage=0,motorOverTemp=0,moduleOverT...	0x00008926@Data
motorVars_M1.speed_int_Hz	float	0.0	0x000089A6@Data
motorVars_M1.angleFOC_rad	float	0.0	0x000089AE@Data
motorVars_M1.angleEST_rad	float	0.0	0x000089B0@Data
motorVars_M1.angleGen_rad	float	0.0	0x000089B4@Data
motorCtrlVars_M1.flagEnableRunMotor	unsigned char	0 '\x00'	0x000086C4@Data
motorCtrlVars_M1.flagEnableSpeedCtrl	unsigned char	0 '\x00'	0x000086C6@Data
motorCtrlVars_M1.accelerationMax_Hzps	float	0.0	0x000086E0@Data
motorCtrlVars_M1.IqSet_A	float	0.0	0x000086D8@Data
motorVars_M1.speedEST_Hz	float	0.0	0x000089AA@Data
motorVars_M1.speed_Hz	float	0.0	0x000089AC@Data
motorVars_M1.angleFOC_rad	float	0.0	0x000089AE@Data
motorVars_M1.angleEST_rad	float	0.0	0x000089B0@Data
motorVars_M1.anglePLL_rad	float	0.0	0x000089B2@Data
motorVars_M1.RsOnLine_Ohm	float	0.0	0x00008994@Data
motorSetVars_M1.Rs_Ohm	float	0.0	0x00008790@Data
motorSetVars_M1.Ls_d_H	float	0.0	0x00008792@Data
motorSetVars_M1.Ls_q_H	float	0.0	0x00008794@Data
motorSetVars_M1.flux_VpHz	float	0.0	0x00008796@Data
motorVars_M1.Rs_Ohm	float	0.0	0x00008984@Data
motorVars_M1.Ls_d_H	float	0.0	0x00008986@Data
motorVars_M1.Ls_q_H	float	0.0	0x00008988@Data
motorVars_M1.flux_VpHz	float	0.0	0x0000898A@Data

图 3-24. 构建级别 1：重置时的“Expressions”监视窗口

3.3.4.1.4 运行代码

要运行代码，请完成以下步骤：

1. 通过点击按钮  来运行工程，或点击 *Debug* 选项卡中的 *Run* → *Resume*。
2. 在“Watch”窗口中看到 *systemVars.flagEnableSystem* 自动设置为“1”后，在 *Expressions* 窗口中，将 *motorVars_M1.flagEnableRunAndIdentify* 设置为“1”。
3. 工程现在可以运行，在使用该工程期间，图和 *Expressions* 窗口中的值可以不断更新，如图 3-25 所示。用户可以根据自己的偏好调整这些窗口的大小。
4. 在监视视图中，变量 *motorVars_M1.flagRunIdentAndOnLine* 可以自动设置为“1”。*ISRCount* 会不断增加。
5. 检查电机的校准偏移，电机相电流检测的偏移值可以约等于 ADC 标度电流的一半，如图 3-25 所示。

6. 在 J15 处使用示波器探测电机驱动控制的 PWM 输出，如图 3-23 中所示。在该构建级别中，所有 PWM 占空比都设置为 50%，PWM 输出波形如图 3-26 所示。motor_1 的 PWM 开关频率为 15kHz。
7. 现在可以停止控制器，并终止调试连接。通过首先点击工具栏上的 *Halt* 按钮 或点击 *Target* → *Halt* 来完全停止控制器。最后，通过点击 或点击 *Run* → *Reset* 来重置控制器。
8. 通过点击 *Tools* → *On-Chip Flash*，然后点击 *On-Chip Flash* 选项卡中的 *Erase Flash* 来擦除控制器中的代码以实现下一个构建级别（确保选中了所有闪存组），如图 3-27 所示。该操作会擦除闪存中存储的所有程序代码。（该步骤是可选的，用户可以忽略该步骤以在下一个构建级别加载新的程序代码。）

备注

在擦除闪存时请勿点击 *Cancel*、关闭板的电源或断开仿真器。

9. 通过点击 *Terminate Debug Session* 按钮 或点击 *Run* → *Terminate* 来关闭 CCS 调试会话。

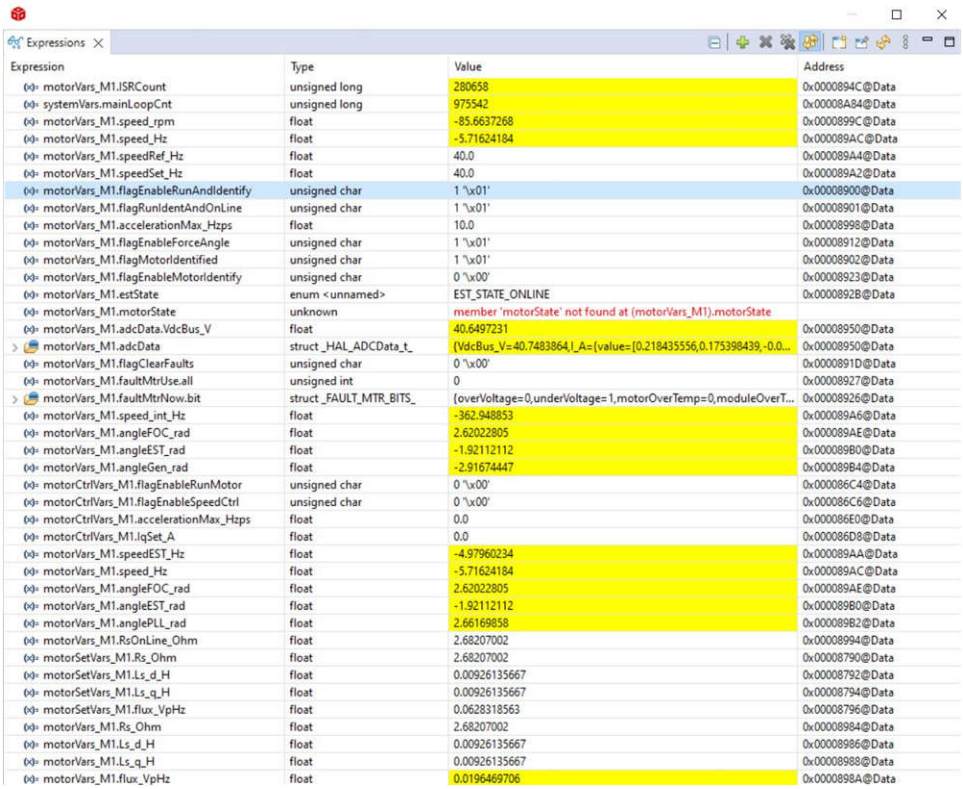


图 3-25. 构建级别 1：运行时的“Expressions”窗口

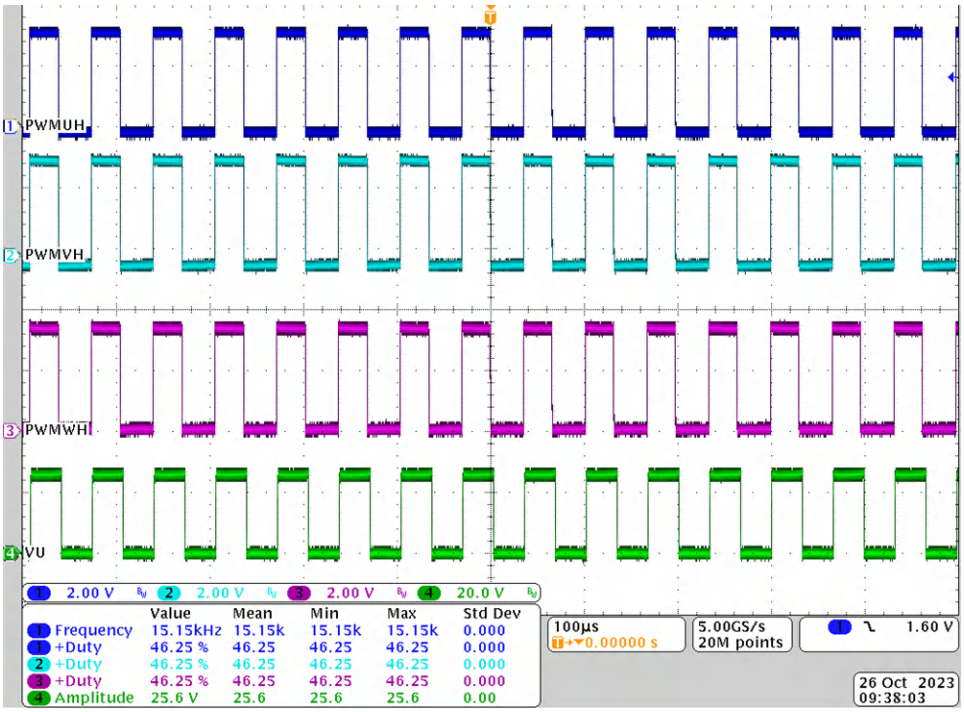


图 3-26. 构建级别 1：MCU PWM 输出和 IPM 输出

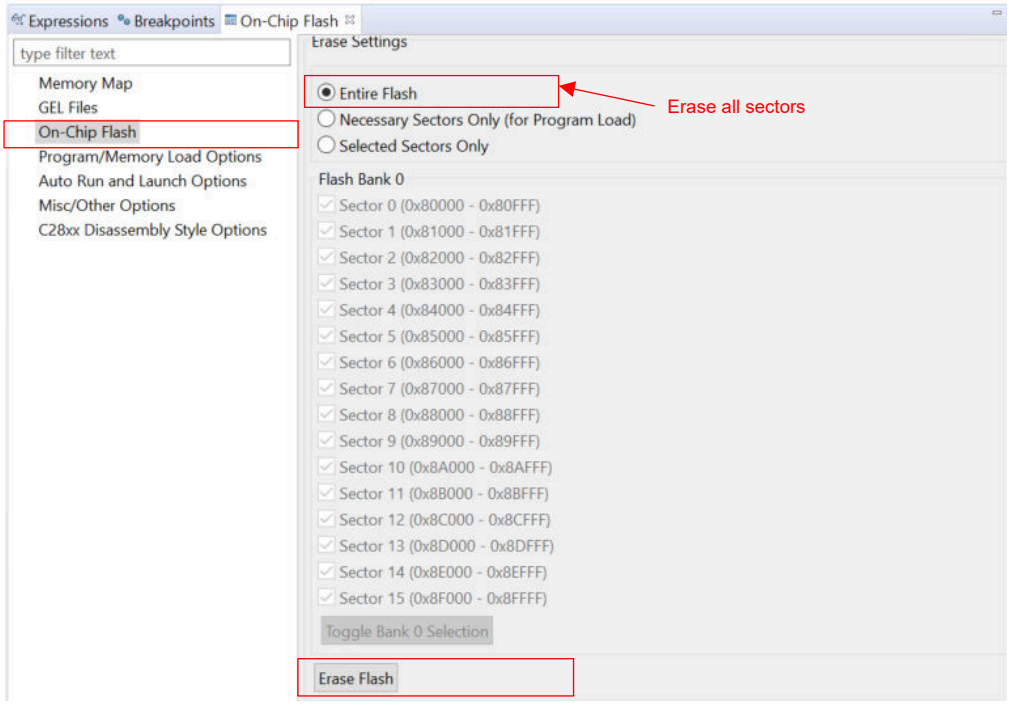


图 3-27. 构建级别 1：擦除闪存中的程序代码以实现下一个构建级别

3.3.4.2 构建级别 2：带 ADC 反馈的开环检查

了解该构建级别中的目标：

- 实现简单的电机标量 v/f 控制以驱动电机，从而验证电流和电压检测电路以及 IPM 电路。
- 测试用于电机控制的 InstaSPIN-FOC FAST 或 eSMO 模块。

由于该系统以开环控制方式运行，因此 ADC 测量值只用于该构建级别的仪器用途。逆变器提供高电压直流电源，辅助电源模块提供 MCU 控制器和 IPM 的偏置电源。图 3-28 展示了该构建级别的软件流程图。

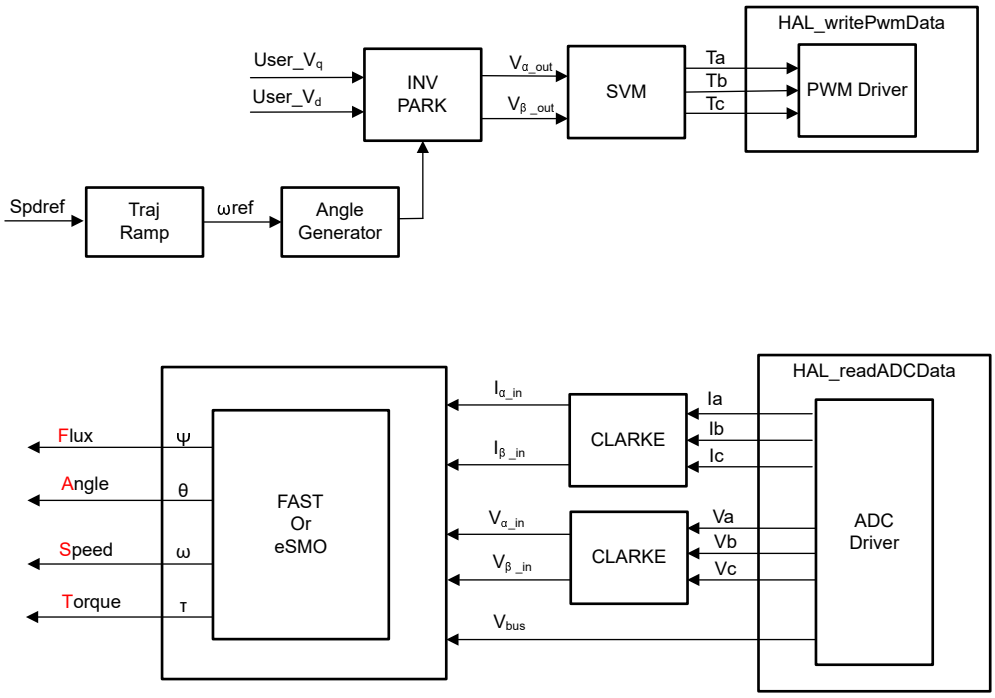


图 3-28. 控制软件方框图：构建级别 2 – 开环控制

3.3.4.2.1 启动 CCS 并打开工程

要启动 CCS 并打开工程，请完成以下步骤：

1. 将一个能够提供高达 750W 通用输入的隔离式交流电源连接至参考板的输入端子（连接器 J5），如图 3-23 所示。将电源电流限制设置为 2A。此时不要打开电源。
2. 将电机连接到 J10。
3. 按照节 3.3.4.1.1 中的步骤 2 和 3 打开工程。

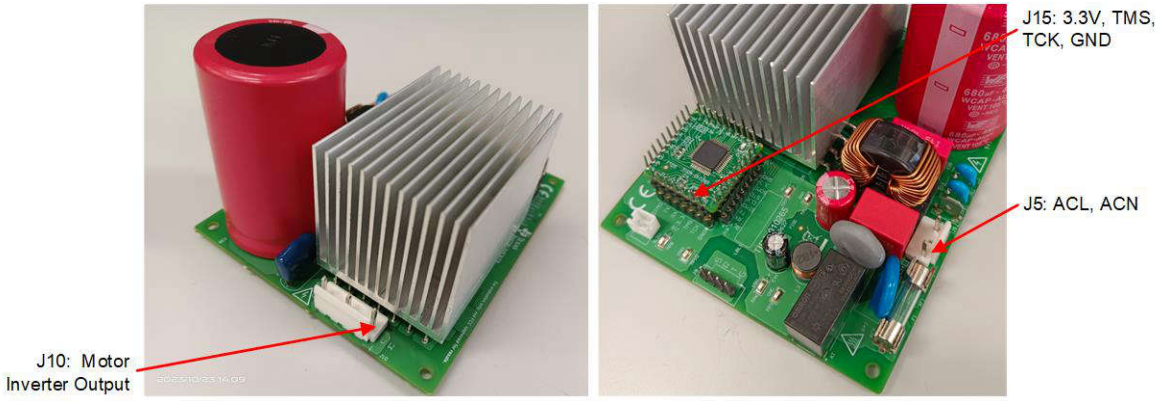


图 3-29. 连接外部交流或直流电源以验证硬件

3.3.4.2.2 构建和加载工程

要构建和加载工程，请完成以下步骤：

1. 将 DMC_BUILDLEVEL 设置为 DMC_LEVEL_2。
2. 按照节 3.3.4.1.2 中的步骤 2 至 4 构建工程并将代码加载到控制器中。

3.3.4.2.3 设置调试环境窗口

通过选择 *BuildLevel2.txt*，按照节 3.3.4.1.3 中的步骤 1 至 4 将变量导入 *Expressions* 窗口。此时将显示 *Expressions* 窗口，如图 3-30 所示。

3.3.4.2.4 运行代码

要运行代码，请完成以下步骤：

1. 将交流电源输出设置为 0V，打开交流电源，将输出电压从 0V 缓慢增加至 100V 交流。
2. 通过点击 按钮来运行工程，或点击 *Debug* 选项卡中的 *Run* → *Resume*。电机故障标志 *motorVars_M1.faultMtrUse.all* 需要等于“0”，否则用户必须检查电流和电压检测电路，如节 3.3.4.1 中所述。
3. 要验证电机逆变器的电流和电压检测电路，请在 *Expressions* 窗口中将变量 *motorVars_M1.flagEnableRunAndIdentify* 设置为“1”，如图 3-30 所示。motor_1 需要以 v/f 开环运行，如果电机旋转不平稳，请根据电机规格调整 *user_mtr1.h* 中的 v/f 曲线参数，如下所示。

```
#define USER_MOTOR1_FREQ_LOW_Hz              (10.0f)          // Hz
#define USER_MOTOR1_FREQ_HIGH_Hz             (200.0f)         // Hz
#define USER_MOTOR1_VOLT_MIN_V               (10.0f)          // Volt
#define USER_MOTOR1_VOLT_MAX_V               (200.0f)         // Volt
```

4. 电机现在以变量 *motorVars_M1.speedRef_Hz* 中设置的速度旋转，并在 *Expressions* 窗口中检查 *motorVars_M1.speed_Hz* 的值。该值需要非常接近，如图 3-30 所示。
5. 如图 3-31 所示，连接示波器电压和电流探头以观察电机相电压和电流。
6. 通过减小变量 *motorVars_M1.overCurrent_A* 的值来验证过流故障保护，过流保护由 CMPSS 模块实现。如果 *motorVars_M1.overCurrent_A* 设为小于实际电流的值，则会触发过流故障，同时将禁用 PWM 输出，*motorVars_M1.flagEnableRunAndIdentify* 将清除为 0，*motorVars_M1.faultMtrUse.all* 将设置为“0x10”。
7. 现在可以停止控制器，并终止调试连接。通过首先点击工具栏上的 *Halt* 按钮 或点击 *Target* → *Halt* 来完全停止控制器。最后，通过点击 或点击 *Run* → *Reset* 来重置控制器。
8. 通过点击 *Terminate Debug Session* 或点击 *Run* → *Terminate* 来关闭 CCS 调试会话。

Expressions

Expression	Type	Value	Address
motorVars_M1.ISRCount	unsigned long	544363	0x0000894C@Data
systemVars.mainLoopCnt	unsigned long	2059177	0x00008A84@Data
motorVars_M1.speed_rpm	float	1176.95312	0x0000899C@Data
motorVars_M1.speed_Hz	float	80.0597916	0x000089AC@Data
motorVars_M1.speedRef_Hz	float	80.0	0x000089A4@Data
motorVars_M1.speedSet_Hz	float	80.0	0x000089A2@Data
motorVars_M1.flagEnableRunAndIdentify	unsigned char	1 '\x01'	0x00008900@Data
motorVars_M1.flagRunIdentAndOnLine	unsigned char	1 '\x01'	0x00008901@Data
motorVars_M1.accelerationMax_Hzps	float	20.0	0x00008998@Data
motorVars_M1.flagEnableForceAngle	unsigned char	1 '\x01'	0x00008912@Data
motorVars_M1.flagMotorIdentified	unsigned char	1 '\x01'	0x00008902@Data
motorVars_M1.flagEnableMotorIdentify	unsigned char	0 '\x00'	0x00008923@Data
motorVars_M1.estState	enum <unnamed>	EST_STATE_ONLINE	0x0000892B@Data
motorVars_M1.motorState	unknown	member 'motorState' not found at (motorVars_M1).motorState	
motorVars_M1.adcData.VdcBus_V	float	141.188721	0x00008950@Data
motorVars_M1.adcData	struct _HAL_ADCData_t_	{VdcBus_V=136.650162,I_A={value={2.74908686,3.78245902,4.2660...	0x00008950@Data
motorVars_M1.flagClearFaults	unsigned char	0 '\x00'	0x0000891D@Data
motorVars_M1.faultMtrUse.all	unsigned int	0	0x00008927@Data
motorVars_M1.faultMtrNow.bit	struct _FAULT_MTR_BITS_	{overVoltage=0,underVoltage=0,motorOverTemp=0,moduleOverT...	0x00008926@Data
motorVars_M1.speed_int_Hz	float	80.0	0x000089A6@Data
motorVars_M1.angleFOC_rad	float	-1.86383724	0x000089AE@Data
motorVars_M1.angleEST_rad	float	0.0245006047	0x000089B0@Data
motorVars_M1.angleGen_rad	float	1.99211061	0x000089B4@Data
motorCtrlVars_M1.flagEnableRunMotor	unsigned char	0 '\x00'	0x000086C4@Data
motorCtrlVars_M1.flagEnableSpeedCtrl	unsigned char	0 '\x00'	0x000086C6@Data
motorCtrlVars_M1.accelerationMax_Hzps	float	0.0	0x000086E0@Data
motorCtrlVars_M1.IqSet_A	float	0.0	0x000086D8@Data
motorVars_M1.speedEST_Hz	float	78.2935638	0x000089AA@Data
motorVars_M1.speed_Hz	float	80.0597916	0x000089AC@Data
motorVars_M1.angleFOC_rad	float	-1.86383724	0x000089AE@Data
motorVars_M1.angleEST_rad	float	0.0245006047	0x000089B0@Data
motorVars_M1.anglePLL_rad	float	0.653951049	0x000089B2@Data
motorVars_M1.RsOnLine_Ohm	float	2.68207002	0x00008994@Data
motorSetVars_M1.Rs_Ohm	float	2.68207002	0x00008790@Data
motorSetVars_M1.Ls_d_H	float	0.00926135667	0x00008792@Data
motorSetVars_M1.Ls_q_H	float	0.00926135667	0x00008794@Data
motorSetVars_M1.flux_VpHz	float	0.0628318563	0x00008796@Data
motorVars_M1.Rs_Ohm	float	2.68207002	0x00008984@Data
motorVars_M1.Ls_d_H	float	0.00926135667	0x00008986@Data

图 3-30. 构建级别 2：运行时的“Expressions”窗口

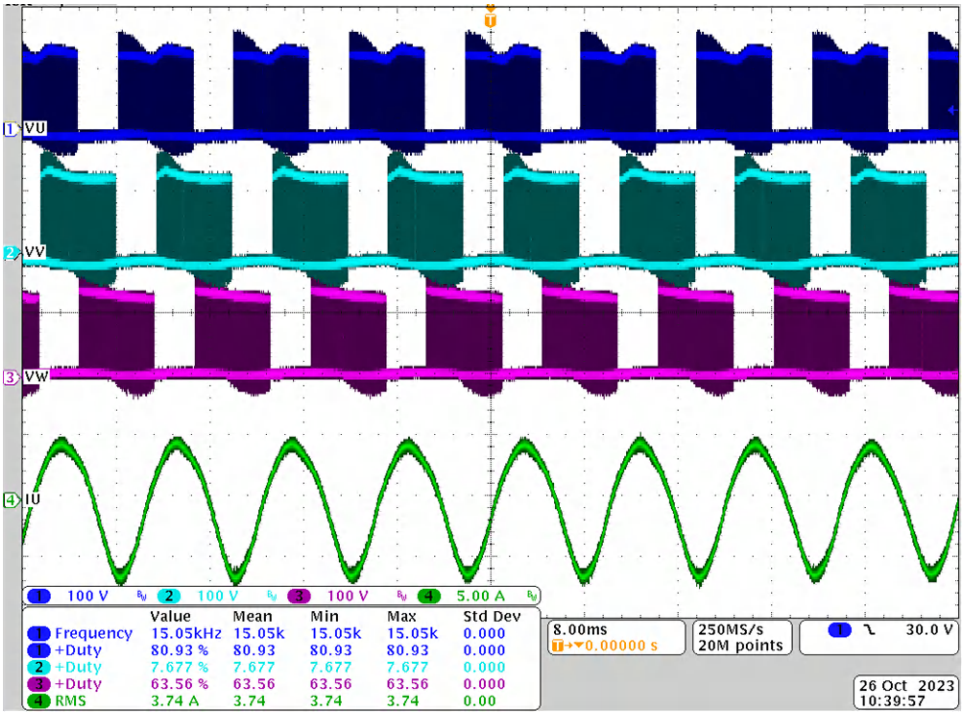


图 3-31. 构建级别 2：电机相电压和电流

3.3.4.3 构建级别 3：闭合电流环路检查

了解该构建级别中的目标：

- 评估电机运行的闭合电流环路。

在这个构建级别，电机由 i/f 控制进行控制，转子角度由斜坡发生器模块生成。图 3-32 展示了该构建级别的软件流程图。

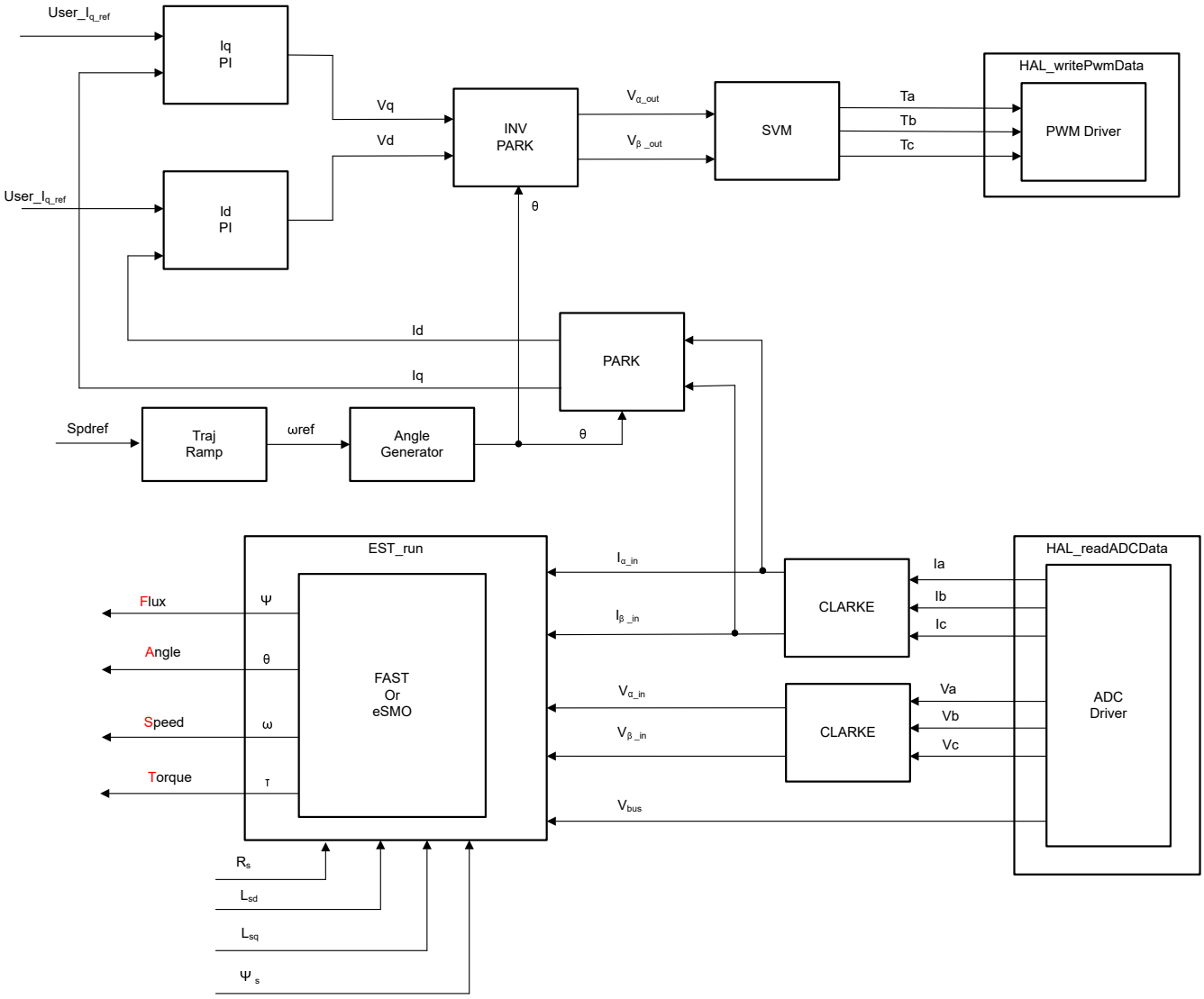


图 3-32. 控制软件方框图：构建级别 3 – 电流闭环控制

3.3.4.3.1 启动 CCS 并打开工程

要启动 CCS 并打开工程，请完成以下步骤：

1. 将一个能够提供高达 750W 通用交流输入的可编程隔离式交流电源连接至参考板的输入端子（连接器 J5），如图 3-29 所示。将电源电流限制设置为 2A，将输出频率设置为 50/60Hz。此时不要打开电源。
2. 将电机（压缩机）连接到 J10。
3. 按照节 3.3.4.1.1 中的步骤 2 和 3 打开工程。

3.3.4.3.2 构建和加载工程

要构建和加载工程，请完成以下步骤：

1. 打开 sys_settings.h 文件，将 DMC_BUILDLEVEL 设置为 DMC_LEVEL_3。

2. 按照节 3.3.4.1.2 中的步骤 2 至 4 构建工程并将代码加载到控制器中。

3.3.4.3.3 设置调试环境窗口

通过选择 *BuildLevel3.txt*，按照节 3.3.4.1.3 中的步骤 1 至 4 将变量导入 *Expressions* 窗口。此时将显示 *Expressions* 窗口，如图 3-30 所示。

3.3.4.3.4 运行代码

要运行代码，请完成以下步骤：

1. 将交流电源输出设置为 0V (50/60Hz)，打开交流电源，将输入电压从 0V 缓慢增加至 220V 交流。
2. 通过点击 按钮来运行工程，或点击 *Debug* 选项卡中的 *Run* → *Resume*。经过固定的时长后，将 *systemVars.flagEnableSystem* 设置为 1，这意味着偏移校准已完成并且浪涌电源继电器已开启。*motorVars_M1.faultMtrUse.all* 的电机故障标志需要等于“0”，否则用户必须检查电流和电压检测电路，如节 3.3.4.1 中所述。
3. 要验证电机的电流闭环控制，请在 *Expressions* 窗口中将变量 *motorVars_M1.flagEnableRunAndIdentify* 设置为“1”，如图 3-33 所示。电机在运行时需要使用来自角度发生器的角度以变量 *motorVars_M1.speedRef_Hz,* 中的设置速度进行闭环控制，检查“Expressions”窗口中 *motorVarsM1.speed_Hz* 的值，这两个变量的值需要非常接近。
4. 电机电流 Iq 可通过 *motorVars_M1.Idq_Set_A.value[1]* 设置和更改
5. 如图 3-34 所示，将示波器探头连接到 IPM 输出以观察电机相电压和电流。更改 *Expressions* 窗口中的 *Idq_set_A[0].value[1]*，电机相电流需要相应地增加。
6. 如果电机无法以电流闭环运行并出现过流故障，则检查是否根据硬件板正确设置了 *adcData[0].current_sf* 的符号和 *userParams[0].current_sf* 的值。
7. 现在可以在将 *motorVars_M1.flagEnableRunAndIdentify* 设置为“0”之前停止控制器，并终止调试连接。通过首先点击工具栏上的 *Halt* 按钮 或点击 *Target* → *Halt* 来完全停止控制器。最后，通过点击 或点击 *Run* → *Reset* 来重置控制器。
8. 通过点击 *Terminate Debug Session* 或点击 *Run* → *Terminate* 来关闭 CCS 调试会话。

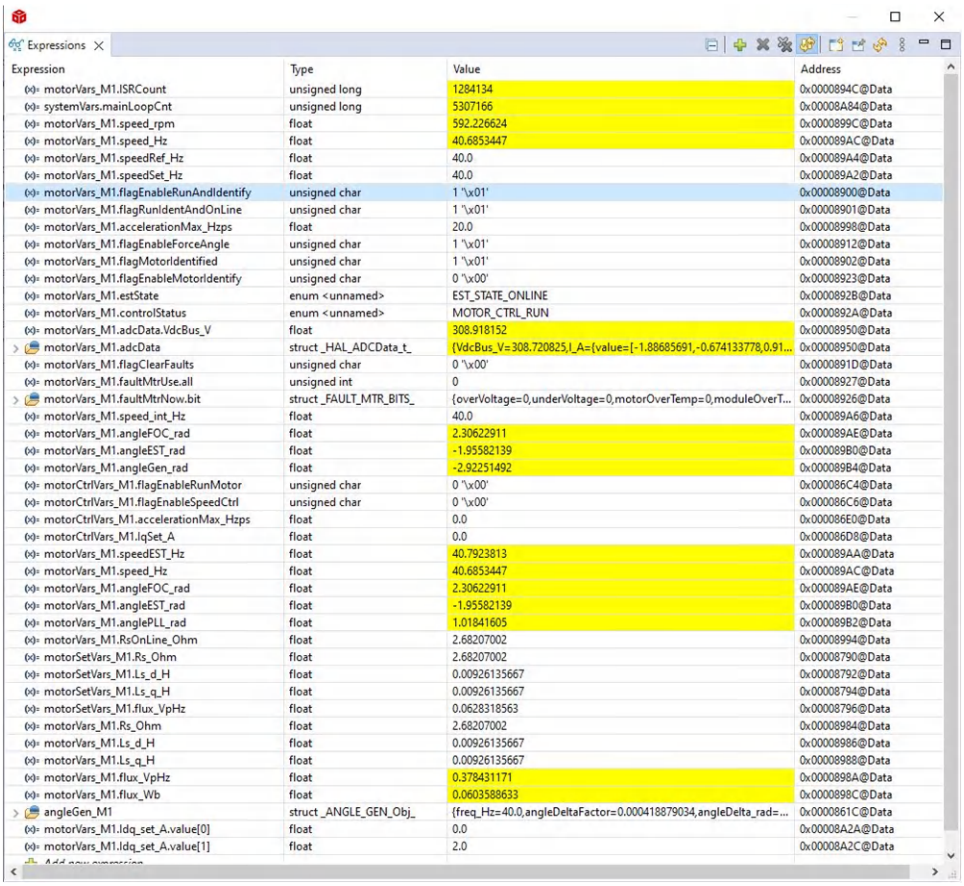


图 3-33. 构建级别 3：运行时的“Expressions”窗口

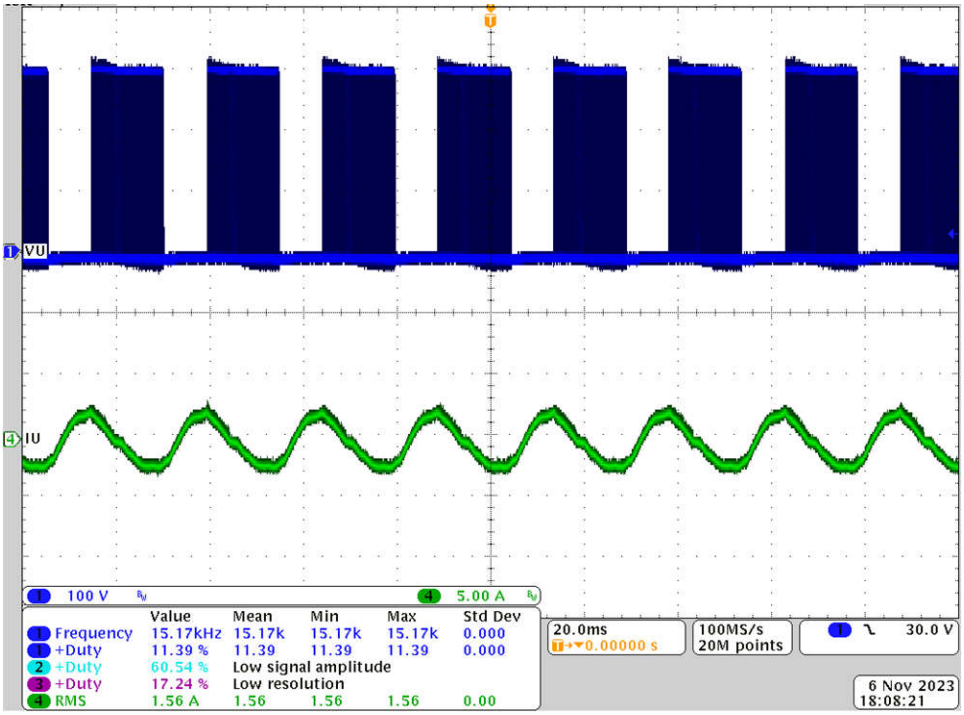


图 3-34. 构建级别 3：2A I_Q 设置下的电机电流

3.3.4.4 版本级别 4：完整电机驱动控制

了解该构建级别中的目标：

- 评估完整的电机驱动
- 评估其他功能，比如电机弱磁控制
- 评估完成的系统。

在此构建级别，外部速度环路是闭合的，内部电流环路用于电机，使得转子角度来自 FAST 或 eSMO 估算器模块。图 3-35 展示了该构建级别的软件流程图。

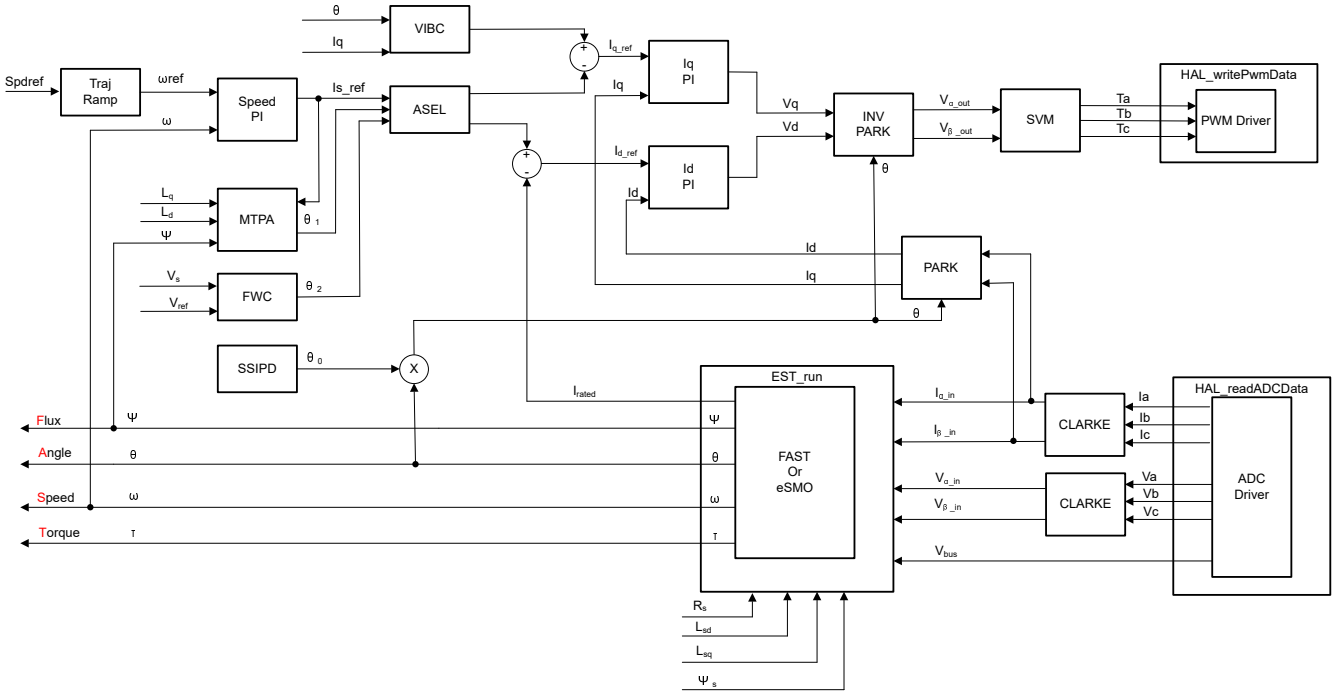


图 3-35. 控制软件方框图：构建级别 4 – 转速和电流闭环控制

3.3.4.4.1 启动 CCS 并打开工程

要启动 CCS 并打开工程，请完成以下步骤：

1. 将一个能够提供高达 750W 通用交流输入的可编程隔离式交流电源连接至参考板的输入端子（连接器 J5），如图 3-29 所示。将交流电源电流限制设置为 8A。此时不要打开电源。
2. 将电机（压缩机）连接到 J10。
3. 按照节 3.3.4.1.1 中的步骤 2 和 3 打开工程。

3.3.4.4.2 构建和加载工程

要构建和加载工程，请完成以下步骤：

1. 打开 sys_settings.h 文件，将 DMC_BUILDLEVEL 设置为 DMC_LEVEL_4。
2. 按照节 3.3.4.1.2 中的步骤 2 至 4 构建工程并将代码加载到控制器中。

3.3.4.4.3 设置调试环境窗口

通过选择 *BuildLevel4.txt*，按照节 3.3.4.1.3 中的步骤 1 至 4 将变量导入 *Expressions* 窗口。此时将显示 *Expressions* 窗口，如图 3-30 所示。

3.3.4.4.4 运行代码

要运行代码，请完成以下步骤：

1. 将交流电源输出设置为 0V (50/60Hz)，打开交流电源，将输入电压从 0V 缓慢增加至 220V 交流。

2. 必须在头文件 (*user_mtr1.h*) 中记录所需的电机参数，如以下示例代码所示。如果用户不太了解电机参数，那么在参考设计中实现了 FAST 估算器的情况下，可以使用电机识别来获得电机参数。

```
#define USER_MOTOR1_Rs_Ohm                  (2.68207002f)
#define USER_MOTOR1_Ls_d_H                  (0.00926135667f)
#define USER_MOTOR1_Ls_q_H                  (0.00926135667f)
#define USER_MOTOR1_RATED_FLUX_VpHz         (0.381890297f)
```

3. 将 *userParams_M1.flag_bypassMotorId* 值更改为“false”以启用电机识别，如以下电机示例代码所示。

```
// true->enable identification, false->disable identification
    userParams[MTR_1].flag_bypassMotorId = false;
```

4. 根据电动的规格在 *user_mtr1.h* 中设置正确的识别变量值。

```
 #define USER_MOTOR1_RES_EST_CURRENT_A       (1.0f)      // A - 10~30% of rated current of the
motor
#define USER_MOTOR1_IND_EST_CURRENT_A       (-1.0f)     // A - 10~30% of rated current of the
motor, just enough to enable rotation
#define USER_MOTOR1_MAX_CURRENT_A           (6.5f)      // A - 30~150% of rated current of the
motor
#define USER_MOTOR1_FLUX_EXC_FREQ_Hz        (40.0f)     // Hz - 10~30% of rated frequency of
the motor
```

5. 重新构建工程并将代码加载到控制器中，通过点击 按钮来运行工程，或点击 *Debug* 选项卡中的 *Run* → *Resume*。经过固定的时长后，*systemVars.flagEnableSystem* 需要设置为“1”，这意味着偏移校准已完成并且浪涌电源继电器已开启。电机故障标志 *motorVars_M1.faultMtrUse.all* 需要等于“0”，否则请按照节 3.3.4.1 中所述检查电流和电压检测电路。
6. 在 *Expressions* 窗口中将变量 *motorVars_M1.flagEnableRunAndIdentify* 设置为“1”（如图 3-36 所示），此时可以执行电机识别，整个过程需要大约 150s。*motorVars_M1.flagEnableRunAndIdentify* 等于“0”后，就表明已识别了电机参数。使用 user_mtr1.h 中新定义的电机参数记录监视窗口值，如下所示：
   - USER_MOTOR1_Rs = motorVars_M1.Rs_Ohm 的值
   - USER_MOTOR1_Ls_d = motorVars_M1.Ls_d_H 的值
   - USER_MOTOR1_Ls_q = motorVars_M1.Ls_q_H 的值
   - USER_MOTOR_RATED_FLUX = motorVars_M1.flux_VpHz 的值
7. 成功识别电机参数后，将 *userParams_M1.flag_bypassMotorId* 设置为“true”，重新构建工程并将代码加载到控制器中。
   - 再次将变量 *motorVars_M1.flagEnableRunAndIdentify* 设为“1”以开始运行电机。
   - 将变量 *motorVars_M1.speedRef_Hz* 设置为不同的值，并观察电机转轴转速的变化情况。
   - 要更改加速度，请为变量 *motorVars_M1.accelerationMax_Hzps* 和 *motorVars_M1.accelerationMax_Hzps* 输入不同的加速度值。
8. 现在可以在将 *motorVars_M1.flagEnableRunAndIdentify* 设置为“0”之前停止控制器，并终止调试连接。通过首先点击工具栏上的 *Halt* 按钮 或点击 *Target* → *Halt* 来完全停止控制器。最后，通过点击 或点击 *Run* → *Reset* 来重置控制器。
9. 通过点击 *Terminate Debug Session* 或点击 *Run* → *Terminate* 来关闭 CCS 调试会话。

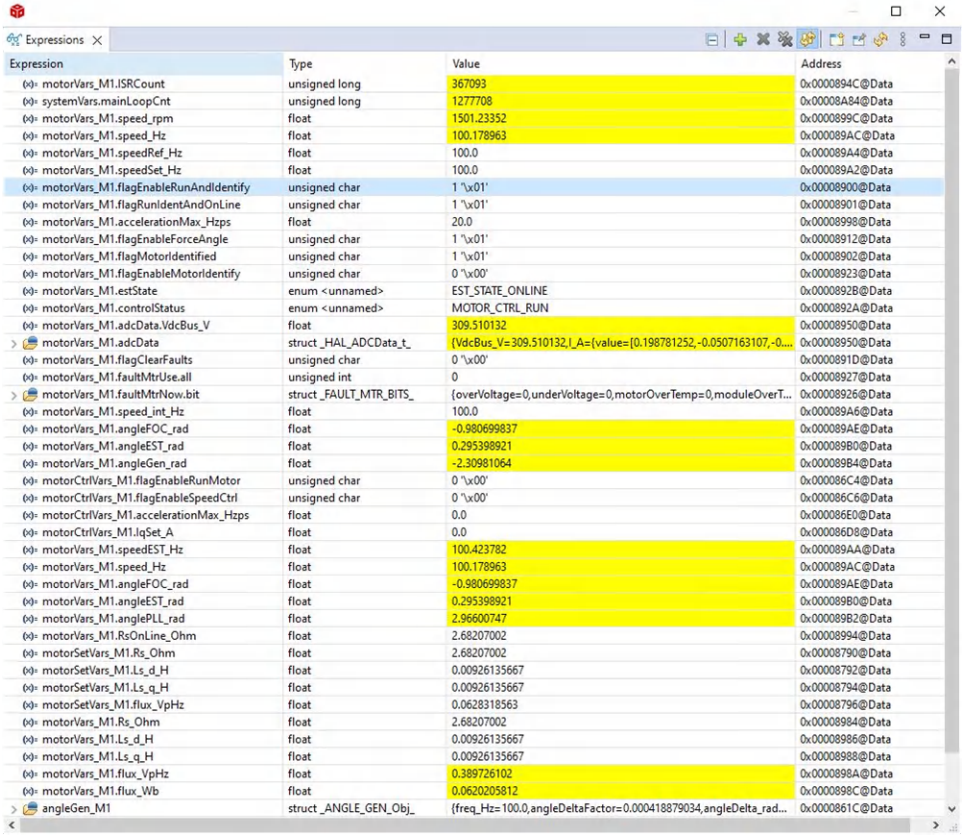


图 3-36. 版本级别 4：运行时的“Expressions”窗口

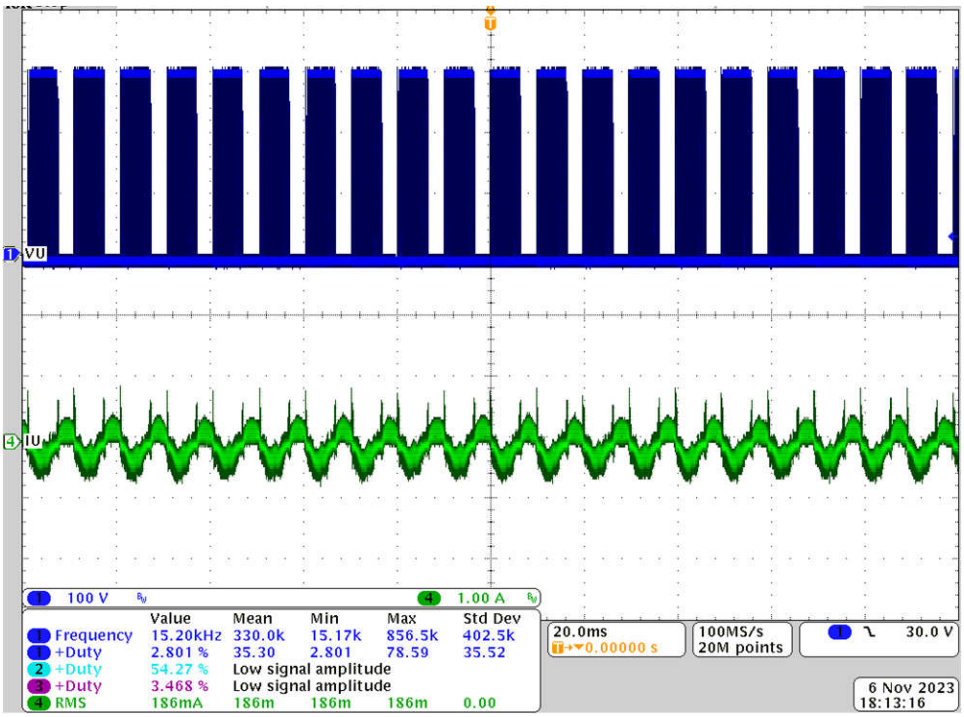


图 3-37. 版本级别 4：电机的转子角度、相电流

3.3.4.4.5 调整电机驱动 FOC 参数

滑模电流观测器由基于模型的电流观测器和继电器式控制发生器组成，后者依靠估算电机电流和实际电机电流之间的误差驱动。F 和 G 参数是根据电机参数 Rs 和 Ls 计算得出的，如节 2.4.2.2.1.2 所述。继电器式控制的观测器增益 k、LPF 的截止频率以及 PLL 角度跟踪器的 K_p 和 K_i 必须根据测试状态进行调整，并尝试获得最佳参数。用户可以并行运行 FAST 估算器和 eSMO，从而验证来自 eSMO 的角度以调整相关参数。初始参数在 user-mtr1.h 文件中进行定义。

```
// Only for eSMO
#define USER_MOTOR1_KSLIDE_MAX              (1.50f)
#define USER_MOTOR1_KSLIDE_MIN              (0.75f)

#define USER_MOTOR1_PLL_KP_MAX              (10.0f)
#define USER_MOTOR1_PLL_KP_MIN              (2.0f)
#define USER_MOTOR1_PLL_KP_SF               (5.0f)

#define USER_MOTOR1_BEMF_THRESHOLD          (0.5f)
#define USER_MOTOR1_BEMF_KSLF_FC_Hz         (2.0f)
#define USER_MOTOR1_THETA_OFFSET_SF         (1.0f)
#define USER_MOTOR1_SPEED_LPF_FC_Hz         (200.0f)
```

可以根据电机参数来计算转速和电流 PI 调节器增益，用户可以在线调整这些增益以优化系统的控制性能。

- 在 CCS Debug 透视图中向 *Expressions* 窗口添加 motorVars[0].Kp_spd、motorVars[0].Ki_spd、motorVars[0].Kp_Iq、motorVars[0].Ki_Iq、motorVars[0].Kp_Id 和 motorVars[0].Ki_Id。更改压缩机电机驱动器的 PI 增益并记录这些值。

3.3.4.4.6 调整弱磁和 MTPA 控制参数

添加 FWC 和 MTPA 函数并在电机驱动器 ISR 中调用来计算电流角，然后计算 d 轴和 q 轴的基准电流。

1. 在工程的构建配置中添加预定义符号 *MOTOR1_FWC* 和 *MOTOR1_MTPA*（如节 3.3.2 所述）以分别启用 FWC 和 MTPA。
2. 在 *user_mtr1.h* 文件中，确保电机参数已知且设置正确。在 *mtpa.h* 中，确保根据电机规格正确设置表格并进行计算。
3. 在 CCS Debug 透视图中向 *Expressions* 窗口添加变量 VsRef_pu、Kp_fwc 和 Ki_fwc，并根据电机及系统调整这些参数以实现弱磁控制的预期性能。
4. 调整并修正这些变量后，使用 *user_mtr1.h* 文件中新定义的参数记录监视窗口值。

   USER_M1_FWC_VREF = VsRef_pu 的值。用于弱磁控制的基准电压系数。

   USER_M1_FWC_KP = Kp_fwc 的值。用于弱磁控制的 PI 稳压器 Kp 增益

   USER_M1_FWC_KI = Ki_fwc 的值。用于弱磁控制的 PI 稳压器 Ki 增益
5. 可以根据电机参数 L_d 、 L_q 以及 ψ_m 来计算 MTPA 控制参数，因此不需要在线调整任何额外的参数。

3.3.4.4.7 调整电流检测参数

精确的电流检测对于估算转子角度和转速以及实现最佳动态电机控制而言非常重要。电流检测参数必须与硬件匹配，这可以通过设置下面的相关参数来实现：

- 死区时间，上升沿延迟时间必须大于功率模块的（高侧开通时间）+（低侧关断时间），下降沿延迟时间必须大于电源模块的（高侧关断时间）+（低侧关断时间），如参考设计中使用的电源模块的以下设置所示。

```
//! \brief Defines the PWM deadband falling edge delay count (system clocks)
#define MTR1_PWM_DBFED_CNT (uint16_t)(2.5f * 120.0f)  // 2.5us, (>2.0us)

//! \brief Defines the PWM deadband rising edge delay count (system clocks)
#define MTR1_PWM_DBRED_CNT (uint16_t)(2.5f * 120.0f)  // 2.50us, (>2.0us)
```

- 脉宽 PWM 的最小持续时间，该时间必须大于（硬件延迟时间 + 死区时间 + 振铃持续时间 + ADC 采样时间）。

```
//! \brief Defines the minimum duration, Clock Cycle
#define USER_M1_DCLINKSS_MIN_DURATION   (450U)
```

- 采样保持延迟时间，它指定从 PWM 输出到 ADC 采样时间的延时时间，用于电流检测。该延迟时间取决于硬件，包括栅极驱动器电路的传播延迟和功率 FET 的导通/关断延迟，其值小于或等于（最小持续时间 - ADC 采样时间）。

```
//! \brief Defines the sample delay, Clock Cycle

#define USER_M1_DCLINKSS_SAMPLE_DELAY   (430U)
```

3.4 测试结果

以下各节介绍了通过表征设计获得的测试数据。测试结果分为多个部分，涵盖风扇和压缩机电机的稳态性能和数据、功能性能波形以及瞬态性能波形。

3.4.1 负载和热力测试

图 3-38 是 500W 无负载条件下 3000RPM (200Hz) 时的波形。该波形显示了以下内容：

- CH1（蓝色）：直流母线电压
- CH2（浅蓝色）：交流输入电压
- CH4（绿色）：U 相电流

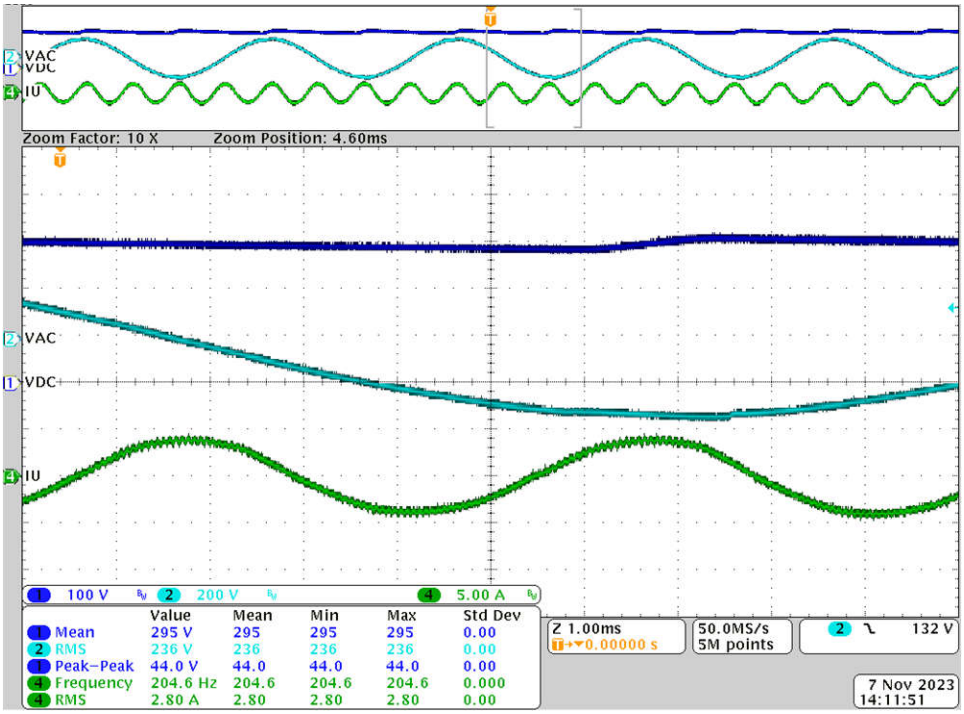


图 3-38. 500W、200Hz 条件下的电机相电流和电压波形

图 3-38 展示了启用磁场减弱时 300W 无负载和 3300RPM (220Hz) 转速条件下的波形。被测电机的额定转速为 3000RPM (200Hz)，现在以弱磁状态工作。

- CH1（蓝色）：直流母线电压
- CH2（浅蓝色）：交流输入电压
- CH4（绿色）：U 相电流

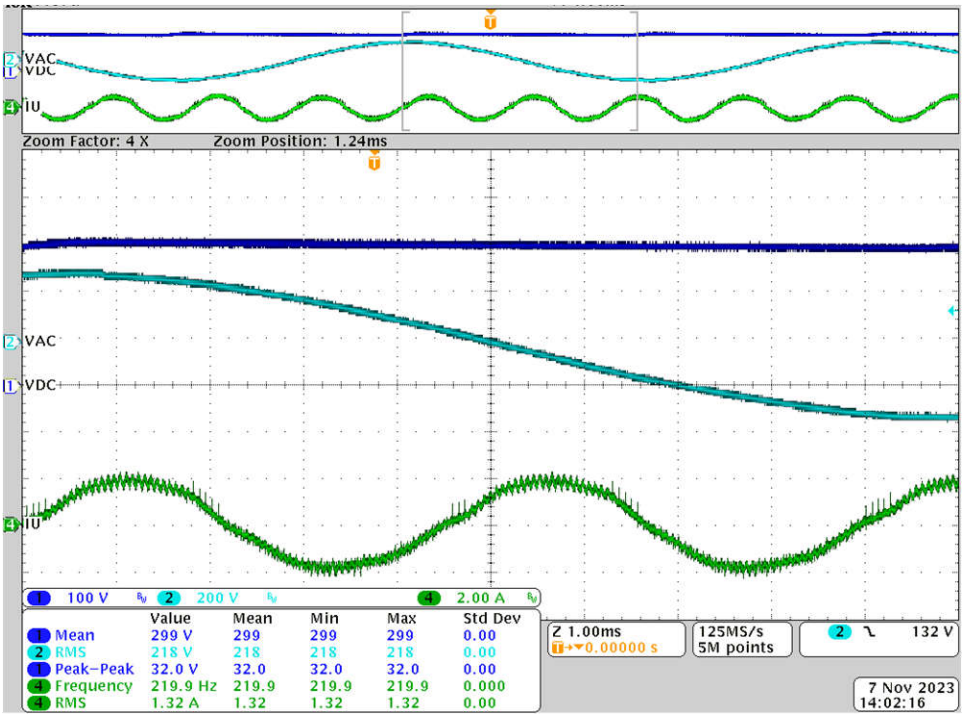


图 3-39. 300W、220Hz 条件下的弱磁测试

该板设计为在 750W 下以短时间（≤1 分钟）工作，并请注意温度升高。如果该电路板以高功率运行或运行时间过长，请使用外部冷却风扇来冷却散热器。图 3-38 展示了 500W、3000RPM (200Hz) 时的电路板温升。

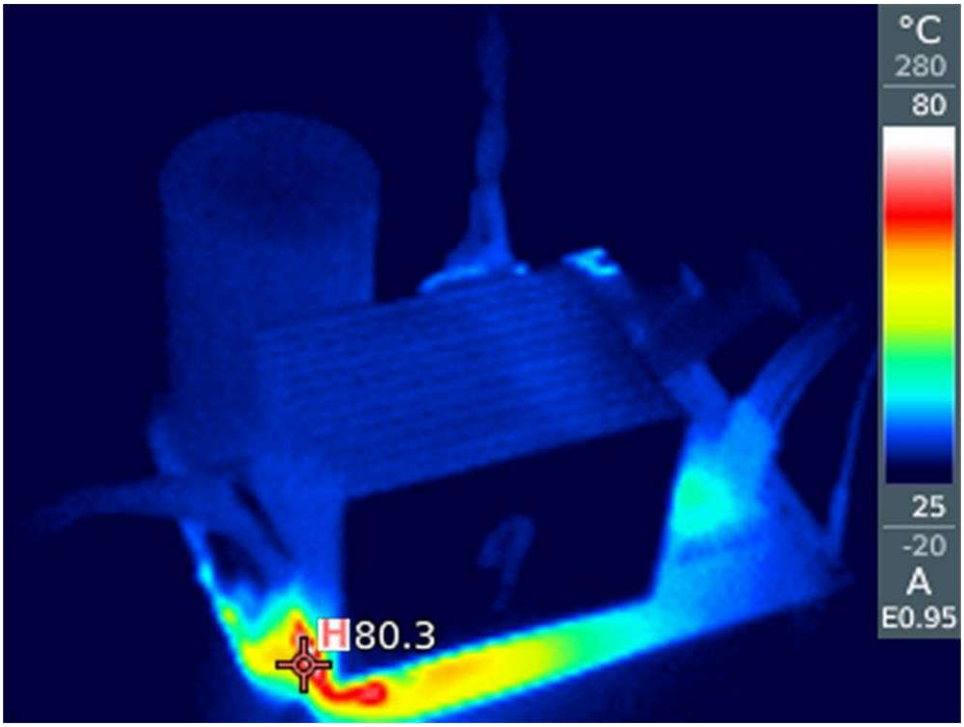


图 3-40. 220V 交流、500W、200Hz 下的热力测试

3.4.2 通过外部比较器进行过流保护

如节 2.4.1.6 中所述，有一个用于提高外部过流保护的比较器 U10。图 3-41 展示了外部过流保护波形。当 R80 上的电流超过 U10 负输入设定的参考点时，U10 的输出 (net IPM_CIN) 为高电平，然后高电平 IPM_CIN 触发 IPM 故障保护，以在 IPM_FAULT 处输出低电平信号，而 IPM_FAULT 连接到微控制器。

- CH1（蓝色）：IPM_FAULT
- CH2（浅蓝色）：IPM_CIN
- CH4（绿色）：R80 的电流

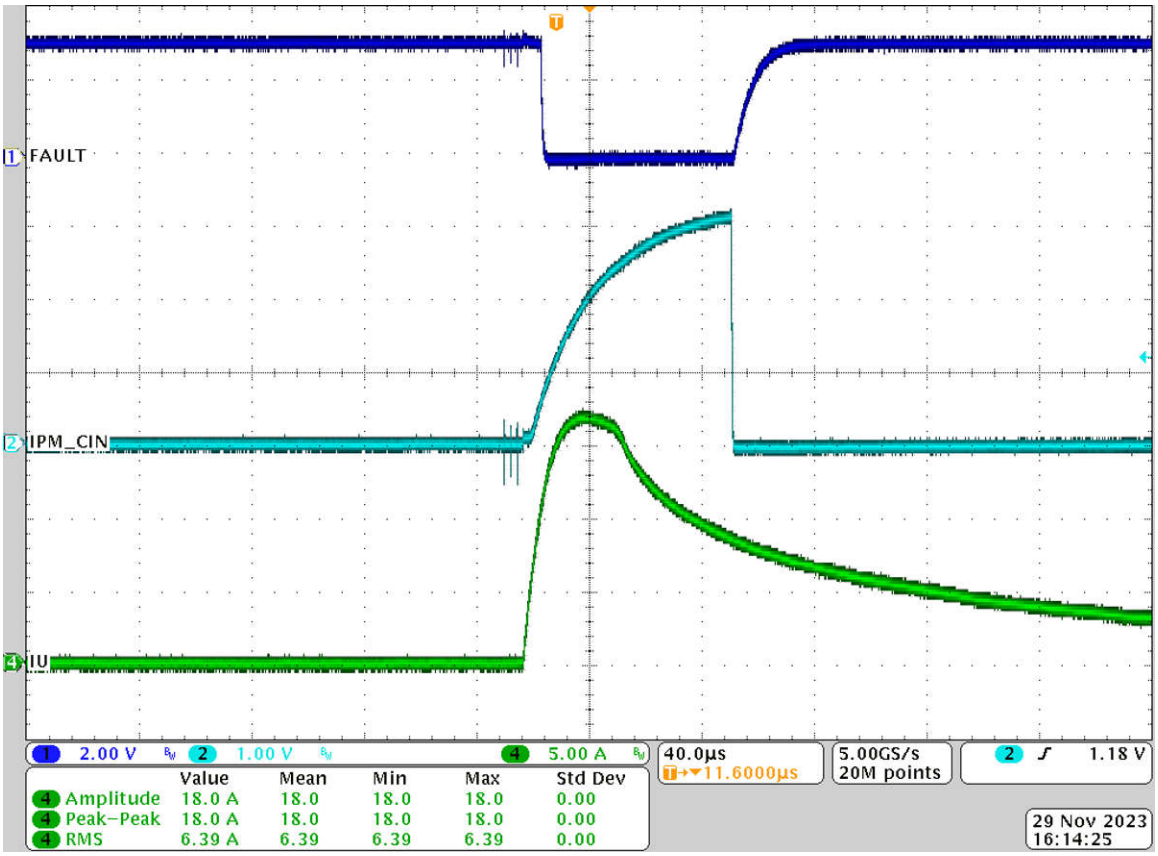


图 3-41. 通过外部比较器进行过流保护

3.4.3 通过内部 CMPSS 进行过流保护

如节 2.4.1.7 中所述，可以配置内部 CMPSS 来提供过流保护。图 3-42 展示了由内部 CMPSS 触发的内部过流保护波形，因为 IPM_FALUT 和 IPM_CIN 均未触发。过流可以通过以下代码设置。

*objSets->maxPeakCurrent_A = USER_M1_ADC_FULL_SCALE_CURRENT_A * 0.4975f;*

- CH1（蓝色）：IPM_FAULT
- CH2（浅蓝色）：IPM_CIN
- CH4（绿色）：U 相电流

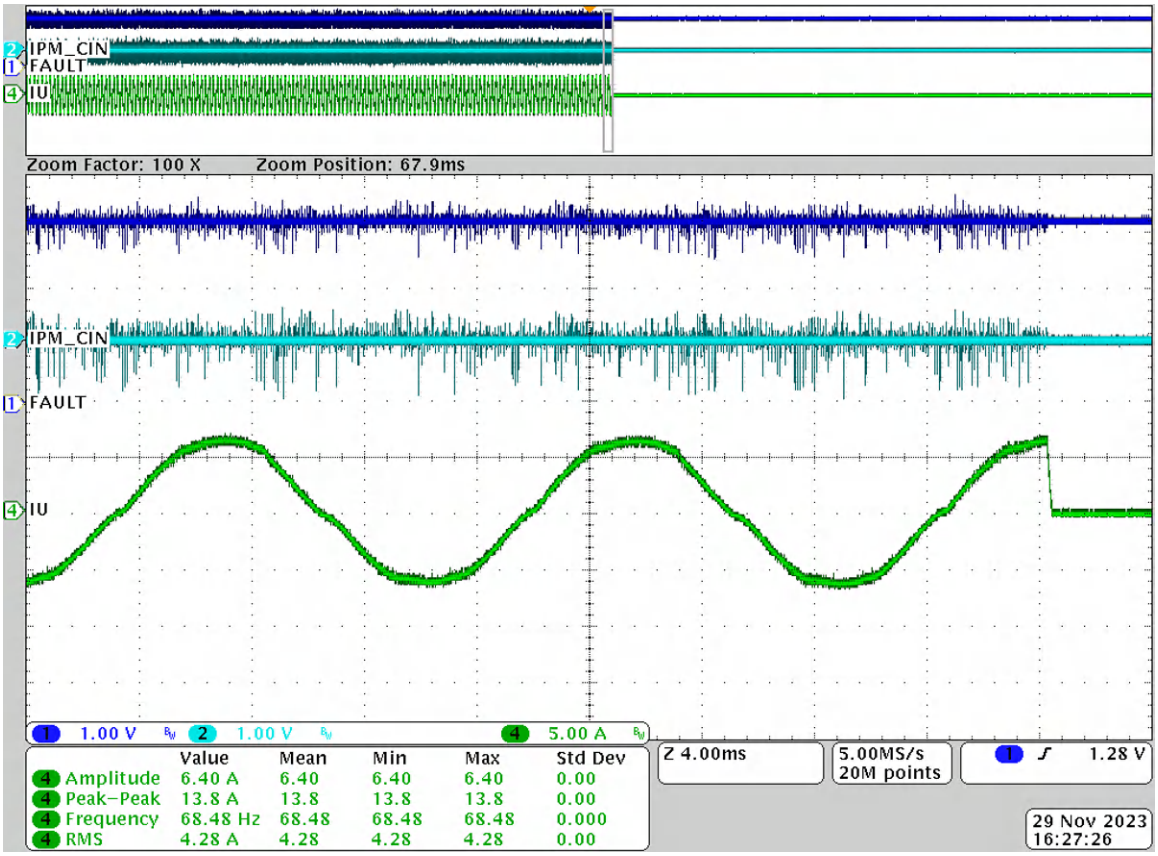


图 3-42. 通过内部 CMPSS 进行过流保护

3.5 将固件迁移至新的硬件板

如果设计人员希望将参考设计迁移到硬件板上，请在 *hal.c*、*hal.h* 和 *user_mtr1.h* 文件中相应地更改与电机相关的 PWM、CMPSS、ADC 外设配置、硬件参数和电机参数，如以下各节所述。

3.5.1 配置 PWM、CMPSS 和 ADC 模块

用于控制电机的应用参数写为 `#define`，可根据硬件，在 *hal.h* 中配置 PWM、CMPSS 和 ADC 模块基地址。压缩机电机定义的 PWM、CMPSS 和 ADC 如以下代码所示。

为电机驱动配置 PWM 和 CMPSS 基地址：

```
// EPWM
#define MTR1_PWM_U_BASE          EPWM2_BASE
#define MTR1_PWM_V_BASE          EPWM3_BASE
#define MTR1_PWM_W_BASE          EPWM4_BASE

// CMPSS->Iu/Iv/Iw
#define MTR1_CMPSS_U_BASE        CMPSSLITE4_BASE
#define MTR1_CMPSS_V_BASE        CMPSSLITE2_BASE
#define MTR1_CMPSS_W_BASE        CMPSSLITE3_BASE
```

为电机驱动配置 ADC 基地址和通道：

```
// Three shunts
// Using ADCA/ADCC for current sensing
#define MTR1_ADC_TRIGGER_SOC     ADC_TRIGGER_EPWM2_SOCA  // EPWM2_SOCA

#define MTR1_ADC_I_SAMPLEWINDOW      14
#define MTR1_ADC_V_SAMPLEWINDOW      20

#define MTR1_IU_ADC_BASE    ADCC_BASE        // ADCC-A7/C3*/CMP4  -SOC1
#define MTR1_IV_ADC_BASE    ADCA_BASE        // ADCA-A4*/C14/CMP2 -SOC2
```

```
#define MTR1_IW_ADC_BASE    ADCA_BASE       // ADCA-A0*/C15/CMP3 -SOC1

#define MTR1_IU_ADCRES_BASE     ADCCRESULT_BASE         // ADCC-A7/C3*
#define MTR1_IV_ADCRES_BASE     ADCARESULT_BASE         // ADCA-A4*/C14
#define MTR1_IW_ADCRES_BASE     ADCARESULT_BASE         // ADCA-A0*/C15

#define MTR1_IU_ADC_CH_NUM      ADC_CH_ADCIN3           // ADCC-A7/C3*
#define MTR1_IV_ADC_CH_NUM      ADC_CH_ADCIN4           // ADCA-A4*/C14
#define MTR1_IW_ADC_CH_NUM      ADC_CH_ADCIN0           // ADCA-A0*/C15

#define MTR1_IU_ADC_SOC_NUM  ADC_SOC_NUMBER1  // ADCC-A7/C3*  -SOC1-PPB1
#define MTR1_IV_ADC_SOC_NUM  ADC_SOC_NUMBER2  // ADCA-A4*/C14 -SOC2-PPB2
#define MTR1_IW_ADC_SOC_NUM  ADC_SOC_NUMBER1  // ADCA-A0*/C15 -SOC1-PPB1

#define MTR1_IU_ADC_PPB_NUM  ADC_PPB_NUMBER1  // ADCC-A7/C3*  -SOC1-PPB1
#define MTR1_IV_ADC_PPB_NUM  ADC_PPB_NUMBER2  // ADCA-A4*/C14 -SOC2-PPB2
#define MTR1_IW_ADC_PPB_NUM  ADC_PPB_NUMBER1  // ADCA-A0*/C15 -SOC1-PPB1
```

为电机驱动控制配置外设中断：

```
// Interrupt
#define MTR1_PWM_INT_BASE  MTR1_PWM_U_BASE       // EPWM1
#define MTR1_ADC_INT_BASE  ADCC_BASE             // ADCC-A11/C0*-SOC4
#define MTR1_ADC_INT_NUM   ADC_INT_NUMBER1       // ADCC_INT1   -SOC4
#define MTR1_ADC_INT_SOC   ADC_SOC_NUMBER4       // ADCC_INT1   -SOC4

#define MTR1_PIE_INT_NUM   INT_ADCC1             // ADCC_INT1   -SOC4
#define MTR1_CPU_INT_NUM   INTERRUPT_CPU_INT1    // ADCC_INT1-CPU_INT1
#define MTR1_INT_ACK_GROUP INTERRUPT_ACK_GROUP1  // ADCC_INT1-CPU_INT1
```

根据硬件，在 *hal.h* 中配置 ADC 引脚与 CMPSS 模块之间的连接；有关详细信息，请参阅 TMS320F280013x 实时微控制器技术参考手册中的表“模拟引脚和内部连接”。

```
// CMPSS->Iu/Iv/Iw
#define MTR1_CMPSS_U_BASE       CMPSSLITE4_BASE
#define MTR1_CMPSS_V_BASE       CMPSSLITE2_BASE
#define MTR1_CMPSS_W_BASE       CMPSSLITE3_BASE

#define MTR1_IU_CMPHP_SEL  ASYSCTL_CMPHPMUX_SELECT_4  // CMPSS4H-A7/C3*
#define MTR1_IU_CMPLP_SEL  ASYSCTL_CMPLPMUX_SELECT_4  // CMPSS4L-A7/C3*
#define MTR1_IV_CMPHP_SEL  ASYSCTL_CMPHPMUX_SELECT_2  // CMPSS2H-A4*/C14
#define MTR1_IV_CMPLP_SEL  ASYSCTL_CMPLPMUX_SELECT_2  // CMPSS2L-A4*/C14

#define MTR1_IW_CMPHP_SEL  ASYSCTL_CMPHPMUX_SELECT_3  // CMPSS3H-A0*/C15
#define MTR1_IW_CMPLP_SEL  ASYSCTL_CMPLPMUX_SELECT_3  // CMPSS3L-A0*/C15

#define MTR1_IU_CMPHP_MUX       1                     // CMPSS4H-A7/C3*
#define MTR1_IU_CMPLP_MUX       1                     // CMPSS4L-A7/C3*

#define MTR1_IV_CMPHP_MUX       0                     // CMPSS2H-A4*/C14
#define MTR1_IV_CMPLP_MUX       0                     // CMPSS2L-A4*/C14

#define MTR1_IW_CMPHP_MUX       2                     // CMPSS3H-A0*/C15
#define MTR1_IW_CMPLP_MUX       2                     // CMPSS3L-A0*/C15
```

根据硬件，在 *hal.h* 中配置从 CMPSS 传递到 EPWM 和 GPIO 输出的跳闸信号，有关详细信息，请参阅 TMS320F280013x 实时微控制器技术参考手册中的“ePWM X-BAR 多路复用器配置表”和“输出 X-BAR 多路复用器配置表”。

```
// XBAR-EPWM
#define MTR1_XBAR_TRIP_ADDRL    XBAR_O_TRIP8MUX0TO15CFG
#define MTR1_XBAR_TRIP_ADDRH    XBAR_O_TRIP8MUX16TO31CFG

#define MTR1_XBAR_INPUT1        XBAR_INPUT1
#define MTR1_TZ_OSHT1           EPWM_TZ_SIGNAL_OSHT1

#define MTR1_XBAR_TRIP          XBAR_TRIP8
#define MTR1_DCTRIPIN           EPWM_DC_COMBINATIONAL_TRIPIN8y

// XBAR-EPWM->Iu/Iv/Iw
#define MTR1_IU_XBAR_EPWM_MUX   XBAR_EPWM_MUX06_CMPSS4_CTRIPH_OR_L      // CMPSS4-HP&LP, A7/C3*
#define MTR1_IV_XBAR_EPWM_MUX   XBAR_EPWM_MUX02_CMPSS2_CTRIPH_OR_L      // CMPSS2-HP&LP, A4*/C14
#define MTR1_IW_XBAR_EPWM_MUX   XBAR_EPWM_MUX04_CMPSS3_CTRIPH_OR_L      // CMPSS3-HP&LP, A0*/C15
```

```
#define MTR1_IU_XBAR_MUX        XBAR_MUX06     // CMPSS4-HP&LP, A7/C3*
#define MTR1_IV_XBAR_MUX        XBAR_MUX02     // CMPSS2-HP&LP, A4*/C14
#define MTR1_IW_XBAR_MUX        XBAR_MUX04     // CMPSS3-HP&LP, A0*/C15
```

相关的 ADC 通道用于电机电流检测哪些引脚在内部连接到比较器子系统 (CMPSS)，在 *hal.c* 文件中的 HAL_setupCMPSSs() 函数中配置 CMPSS 寄存器，代码如下所示。三个 CMPSS 模块用于实现电机 U 相、V 相和 W 相的正负过流保护。

```
void HAL_setupCMPSSsMTR(HAL_MTR_Handle handle)
{
    HAL_MTR_Obj *obj = (HAL_MTR_Obj *)handle;

#if defined(DMCPFC_REV3P2) || defined(DMCPFC_REV3P1)
#if !defined(MOTOR1_DCLINKSS) || !defined(MOTOR2_DCLINKSS)
    uint16_t cmpsaDACH;
#endif  // !(MOTOR1_DCLINKSS || MOTOR2_DCLINKSS)
    uint16_t cmpsaDACL;
    ... ...
#else   // !MOTOR1_DCLINKSS, Three-shunt
        cmpsaDACH = MTR1_CMPSS_DACH_VALUE;
        cmpsaDACL = MTR1_CMPSS_DACL_VALUE;

        ASysCtl_selectCMPHPMux(MTR1_IU_CMPHP_SEL, MTR1_IU_CMPHP_MUX);

        ASysCtl_selectCMPHPMux(MTR1_IV_CMPHP_SEL, MTR1_IV_CMPHP_MUX);

        ASysCtl_selectCMPLPMux(MTR1_IW_CMPLP_SEL, MTR1_IW_CMPLP_MUX);
    ... ...
    return;
} // end of HAL_setupCMPSSs() function
```

CMPSS 生成的信号进入 X-Bar，在此处这些信号能够以不同且独特的方式组合，以标记来自多个来源的独特跳闸事件（包括来自 IPM #Fault 的外部 TZ 信号），从而实现故障保护。故障包括来自 CMPSS 的过流信号和电源模块的故障指示灯输出。在 *hal.c* 文件中的 HAL_setupMtrFaults() 函数中配置 XBAR 寄存器，代码如下所示。

```
void HAL_setupMtrFaults(HAL_MTR_Handle handle)
{
    HAL_MTR_Obj *obj = (HAL_MTR_Obj *)handle;
    uint16_t cnt;

    // Configure TRIP 7 to OR the High and Low trips from both
    // comparator 5, 3 & 1, clear everything first
    EALLOW;
    HWREG(XBAR_EPWM_CFG_REG_BASE + MTR1_XBAR_TRIP_ADDRL) = 0;
    HWREG(XBAR_EPWM_CFG_REG_BASE + MTR1_XBAR_TRIP_ADDRH) = 0;
    EDIS;
    ... ...
        // What do we want the OST/CBC events to do?
        // TZA events can force EPWMxA
        // TZB events can force EPWMxB
        EPWM_setTripZoneAction(obj->pwmHandle[cnt],
                               EPWM_TZ_ACTION_EVENT_TZA,
                               EPWM_TZ_ACTION_LOW);

        EPWM_setTripZoneAction(obj->pwmHandle[cnt],
                               EPWM_TZ_ACTION_EVENT_TZB,
                               EPWM_TZ_ACTION_LOW);
    ... ...
    // Clear any spurious fault
    EPWM_clearTripZoneFlag(obj->pwmHandle[0], HAL_TZFLAG_INTERRUPT_ALL);
    EPWM_clearTripZoneFlag(obj->pwmHandle[1], HAL_TZFLAG_INTERRUPT_ALL);
    EPWM_clearTripZoneFlag(obj->pwmHandle[2], HAL_TZFLAG_INTERRUPT_ALL);

    return;
}
```

在 *hal.c* 文件中的 HAL_setupGPIOs() 中根据硬件配置 GPIO，代码如下所示。

```
void HAL_setupGPIOs(HAL_Handle handle)
{
    ... ...
```

```
    // GPIO2->EPWM2A->M1_UH
    GPIO_setPinConfig(GPIO_2_EPWM2_A);
    GPIO_writePin(2, 0);
    GPIO_setDirectionMode(2, GPIO_DIR_MODE_OUT);
    GPIO_setPadConfig(2, GPIO_PIN_TYPE_STD);

    // GPIO3->EPWM2B->M1_UL
    GPIO_setPinConfig(GPIO_3_EPWM2_B);
    GPIO_writePin(3, 0);
    GPIO_setDirectionMode(3, GPIO_DIR_MODE_OUT);
    GPIO_setPadConfig(3, GPIO_PIN_TYPE_STD);
    ... ...
    return;
}  // end of HAL_setupGPIOs() function
```

需要根据用于电机控制的 CMPSS，在 *hal.h* 文件中的 HAL_enableMtrPWM() 和 HAL_clearMtrFaultStatus() 中更改配置代码，如下面标记为**粗体**的代码所示。

```
static inline void HAL_enableMtrPWM(HAL_MTR_Handle handle)
{
    HAL_MTR_Obj *obj = (HAL_MTR_Obj *)handle;

    obj->flagEnablePWM = true;

#if defined(DMCPFC_REV3P2) || defined(DMCPFC_REV3P1)
    if(obj->motorNum == MTR_1)
    {
#if defined(MOTOR1_DCLINKSS)
        // Clear any comparator digital filter output latch
        CMPSS_clearFilterLatchLow(obj->cmpssHandle[0]);
#else   // !MOTOR1_DCLINKSS
        // Clear any comparator digital filter output latch
        CMPSS_clearFilterLatchHigh(obj->cmpssHandle[0]);

        CMPSS_clearFilterLatchHigh(obj->cmpssHandle[1]);

        CMPSS_clearFilterLatchLow(obj->cmpssHandle[2]);
    ... ...
    return;
} // end of HAL_enableMtrPWM() function
```

```
static inline void HAL_clearMtrFaultStatus(HAL_MTR_Handle handle)
{
    HAL_MTR_Obj *obj = (HAL_MTR_Obj *)handle;
    ... ...
#if defined(HVMTRPFC_REV1P1) || defined(WMINVBRD_REV1P0) || defined(TIDSMPFC_REV3P2)
    // Clear any comparator digital filter output latch
    CMPSS_clearFilterLatchHigh(obj->cmpssHandle[0]);
    CMPSS_clearFilterLatchLow(obj->cmpssHandle[0]);

    CMPSS_clearFilterLatchHigh(obj->cmpssHandle[1]);
    CMPSS_clearFilterLatchLow(obj->cmpssHandle[1]);

    CMPSS_clearFilterLatchHigh(obj->cmpssHandle[2]);
    CMPSS_clearFilterLatchLow(obj->cmpssHandle[2]);
    ... ...
    return;
} // end of HAL_clearMtrFaultStatus() function
```

3.5.2 设置硬件板参数

user_mtr1.h 文件用于存储所有用户参数以进行电机控制。模数转换器输入端的最大相电流和相电压值取决于硬件，并需要基于电流和电压检测以及对 ADC 输入的调节。使用的电流传感器和电压（相）传感器的数量在 *user_mtr1.h* 中定义，具体取决于硬件。

user_mtr1.h 文件中定义了所有可配置参数。可以使用 *Motor_Drive_Parameters_Calculation.xlsx* Microsoft® Excel® 电子表格计算这些参数。该文件包含在 TIDA-010265 存档文件中，位于文件

夹 ..\solutions\tida_010265\docs 下，要计算这些值，请将这些标记为**粗体**的参数复制到 *user_mtr1.h* 中，如以下代码所示。

```
//! \brief Defines the maximum voltage at the AD converter
//  Full scale voltage of AD converter, not the current voltage
#define USER_M1_ADC_FULL_SCALE_VOLTAGE_V         (404.1292683f)

//! \brief Defines the analog voltage filter pole location, Hz
//!
#define USER_M1_VOLTAGE_FILTER_POLE_Hz           (416.3602877f)

//! \brief Defines the maximum current at the AD converter
#define USER_M1_ADC_FULL_SCALE_CURRENT_A         (15.972f)
```

3.5.3 配置故障保护参数

该系统实现了故障管理，包括过流、过压、欠压、失速、过载、启动失败。故障保护参数在 *user_mtr1.h* 中定义，如以下代码所示，具体取决于硬件板、电机和系统。

```
//! \brief motor over current threshold
#define USER_MOTOR1_OVER_CURRENT_A          (6.5f)          // A

//! \brief motor lost phase current threshold
#define USER_M1_LOST_PHASE_CURRENT_A        (0.2f)

//! \brief motor unbalance ratio percent threshold
#define USER_M1_UNBALANCE_RATIO             (0.2f)
```

```
//! \brief DC bus over voltage threshold
#define USER_M1_OVER_VOLTAGE_FAULT_V        (380.0f)

//! \brief DC bus over voltage threshold
#define USER_M1_OVER_VOLTAGE_NORM_V         (350.0f)

//! \brief DC bus under voltage threshold
#define USER_M1_UNDER_VOLTAGE_FAULT_V       (100.0f)
```

3.5.4 设置电机电气参数

user_mtr1.h 中提供的用于 PMSM 和 BLDC 电机的参数如以下代码所示。如果在该电机上实施 FAST 技术或从电机数据表中获取参数，则可以识别电机参数。

```
#define USER_MOTOR1_TYPE                    MOTOR_TYPE_PM
#define USER_MOTOR1_NUM_POLE_PAIRS          (4)
#define USER_MOTOR1_Rr_Ohm                  (0.0f)
#define USER_MOTOR1_Rs_Ohm                  (2.68207002f)
#define USER_MOTOR1_Ls_d_H                  (0.00926135667f)
#define USER_MOTOR1_Ls_q_H                  (0.00926135667f)
#define USER_MOTOR1_RATED_FLUX_VpHz         (0.381890297f)
```

3.6 MSPM0 固件入门

有关 MSPM0G1507 子板固件的信息，请联系当地的 TI 销售代表。

4 设计和文档支持

4.1 设计文件

4.1.1 原理图

要下载原理图，请参阅 TIDA-010265 中的设计文件。

4.1.2 物料清单

要下载物料清单 (BOM)，请参阅 TIDA-010265 中的设计文件。

4.1.3 PCB 布局建议

该参考设计使用具有两层 2oz 铜的 PCB 来实现，并采用底部 SMD 元件放置方式，以节省成本和电路板面积。在设计 PCB 时，需要注意几个重要方面。以下列出了每个块的系统级放置方式和布局。

- 将各个元件（或单个元件）分为高压与低压、高电流与低电流，以及高阻抗与低阻抗组。将低压和高阻抗元件及信号放置在一起并进行布线，例如与微控制器相关的信号和 IPM 的输入侧。使用整个覆铜区来为这些区域提供集成的 GND 平面。交流输入、滤波器和整流器以及 IPM 输出侧是高压、高电流和低阻抗器件及信号，使用更宽的布线或覆铜来提供高电流路径，并还要将它们与上面的低压、高阻抗信号分开，以减少干扰。
- 大功率路径中的元件处于 PCB 的外边缘，并尽可能互相靠近。微控制器放置在中心，以便与所有需要控制的电源块保持最佳距离。引脚进行了合理分配，以尽可能减小控制信号或反馈信号引线距离并尽可能减少模拟信号和数字信号之间的交叉。
- 交流线路保护和 EMI 滤波器
  - 所有交流线路保护元件紧密地放置在一起，尽可能缩短连接路径。在保护和 EMI 滤波器电路周围提供了接地连接保护。
- 电机驱动器
  - 为了满足高纹波要求，电机驱动器应尽可能靠近薄膜电容器和直流母线电容器组放置。
  - 实施了采用四线检测的低侧分流电阻器方法来进行电流检测。使用具有阻抗匹配电阻器的差分对将来自分流电阻器的检测信号连接到运算放大器电路。分流电阻放置在模块附近，并直接连接接地铜平面。
- 辅助电源
  - 辅助电源的 GND 直接独立地连接直流总线电容器组，以将低电流与逆变器的高电流和高频 GND 布线分开。

4.1.4 Altium 工程

要下载 Altium 工程文件，请参阅 TIDA-010265 中的设计文件。

4.1.5 Gerber 文件

要下载 Gerber 文件，请参阅 TIDA-010265 中的设计文件。

4.2 软件文件

通过 CCSTUDIO 下载 Code Composer Studio 集成开发环境。

从 C2000WARE-MOTORCONTROL-SDK 中的设计文件下载特定于 TIDA-010265 硬件的软件设计文件。

4.3 文档支持

1. 德州仪器 (TI)，*TMS320F280013x 微控制器* 数据表
2. 德州仪器 (TI)，*TMS320F280013x 实时微控制器* 技术参考手册
3. 德州仪器 (TI)，*InstaSPIN-FOC 和 InstaSPIN-MOTION* 用户指南
4. 德州仪器 (TI)，*Motor Control SDK 通用工程和实验* 用户指南
5. 德州仪器 (TI)，*C2000™ 软件频率响应分析器 (SFRA) 库和补偿设计器* 用户指南
6. 德州仪器 (TI)，*使用单一直流链路分流器的 PMSM 无传感器 FOC* 应用手册
7. 德州仪器 (TI)，*C2000 SysConfig* 应用手册。

4.4 支持资源

TI E2E™ 中文支持论坛是工程师的重要参考资料，可直接从专家处获得快速、经过验证的解答和设计帮助。搜索现有解答或提出自己的问题，获得所需的快速设计帮助。

链接的内容由各个贡献者“按原样”提供。这些内容并不构成 TI 技术规范，并且不一定反映 TI 的观点；请参阅 TI 的使用条款。

4.5 商标

5 作者简介

HELY ZHANG 是德州仪器 (TI) 的系统应用工程师，负责开发与家用电器相关的电力输送器件和电机逆变器。Hely 于 2002 年获得了安徽理工大学电力电子专业硕士学位，在加入 TI 之前他曾就职于 SolarEdge 和通用电气。

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265