Freescale Semiconductor
应用笔记

Document Number: AN4254
Rev 0, 5/2011

在 Freescale MQX 操作系统下进行电机控制

作者: Libor Prokop
捷克共和国
Roznov

内容

1 简介

本应用笔记介绍了在操作系统（OS）中进行电机控制所面临的挑战。还评估了可行的实现方案，并为在飞思卡尔操作系统 MQX 中加入电机控制应用程序提供了指导。

1.1 电机控制和 MQX

嵌入式系统正变得越来越复杂，嵌入式系统软件编程人员面临的压力也越来越大。在一个复杂的系统中，操作系统必须实时运行多项任务。具体示例包括以太网、USB、SDHC 等。这类任务的其中之一是电机控制，例如直流电机、无刷直流电机、步进电机，甚至是三相正弦电机（例如 PMSM 或交流感应电机）。

电机控制算法要求对控制任务进行精确的时序控制，比如输出信号生成——它根据操作系统（OS）中相反转子位置进行任务调度，延迟极大。因此，将电机控制任务引入操作系统时须额外小心。本文中的电机控制指的是控制交流感应、BLDC、PM 同步或直流 MQX 等电机的过程。

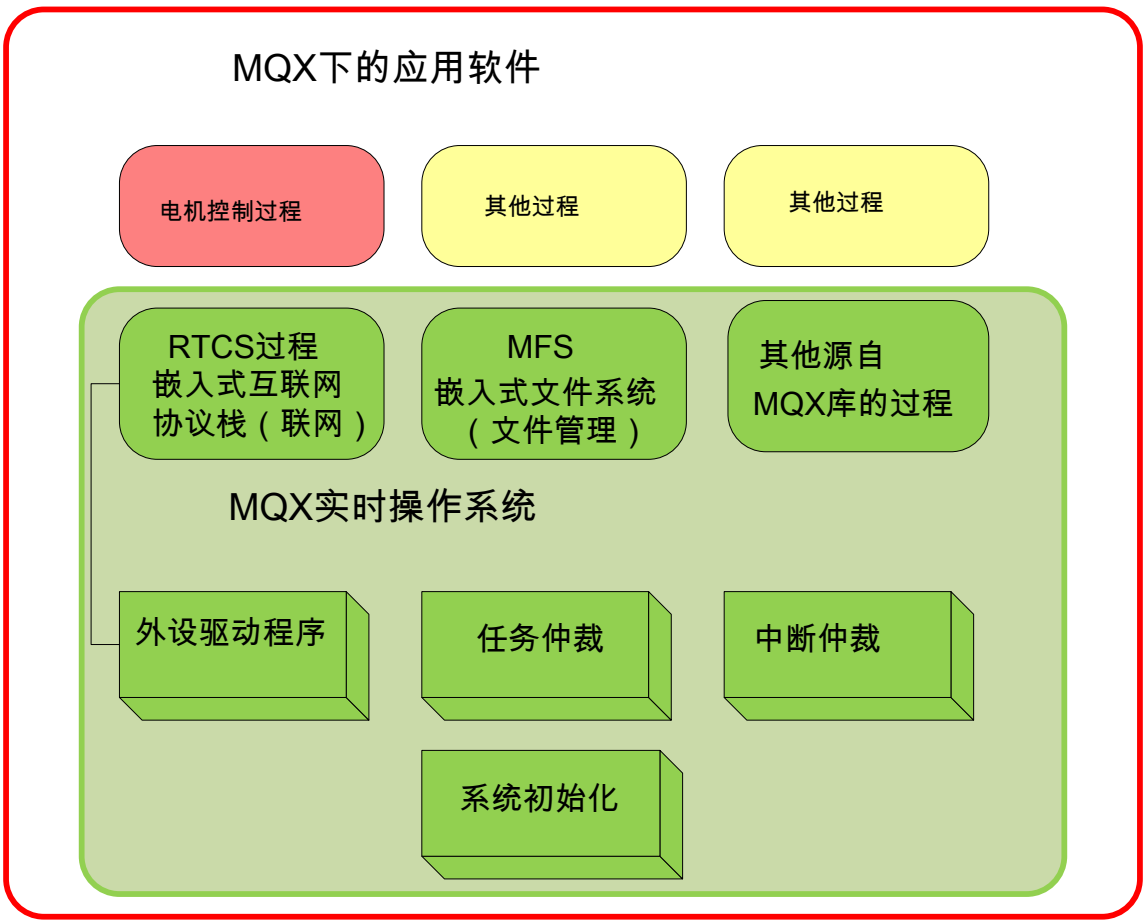

图 1. MQX 以及在 MQX 中进行电机控制的其他应用

1.2 用于高端微控制器的 MQX 实时操作系统

本节说明 MQX 操作系统的基本特性。MQX 是一个实时多任务处理应用程序的运行时函数库。其主要特点是规模可扩展、采用面向组件的架构且易用。MQX 支持多处理器应用程序，可与灵活的嵌入式 I/O 产品配合，用于联网、数据通信、文件管理以及控制。

主要的 MQX 应用领域为大型控制器件，例如 Kinetis (ARM®Cortex™-M4)或 MCF5441x (ColdFire®)系列，这些系列提供以太网、USB、SDHC 等外设和其他支持。部分器件配备 PWM 模块和专为电机控制设计或适用于电机控制的其他外设。MQX 并不是一个专用于电机控制的操作系统，但使用 MQX 操作系统进行电机控制可为一些既有电机控制任务又有其他任务的应用带来好处。

Freescale MQX RTOS 提供完整的软件栈支持，其中包括基本内核驱动程序、类驱动程序以及可用于实现所需目标产品的大量示例程序。MQX 内嵌的 MQX 实时操作系统专为采用单处理器、多处理器和分布式处理器的嵌入式实时系统而设计。为借力 MQX 操作系统的成功，飞思卡尔半导体将这一软件平台应用到其 ColdFire、PowerPC™和 ARM Cortex 微处理器系列中。与原始 MQX 发行版相比，Freescale MQX RTOS 发行版更易于配置和使用。现在，单一发布版本即包含了 MQX 操作系统以及支持指定微处理器器件的所有其他的软件组件。

2 典型电机控制应用

有多种电机类型，它们之间的结构和控制方法各不相同。在 MQX OS 中进行部署时，必须反映这一点。

2.1 电机类型

就电机类型而言，最常见的电机控制应用有:

- 直流电机
  直流电机控制
- 换向电机
  无刷直流（BLDC）电机控制
  步进电机控制
- 步进电机控制
  永磁正弦电机（PMSM）控制
  交流感应电机控制
  步进电机控制

2.2 电机控制技术

根据控制技术划分的关键电机控制应用。

2.2.1 根据传感器的反馈进行控制

- 有传感器的控制

  传感器用于估计转子位置和速度。最常见的传感器类型有:

  - 霍尔传感器
  - 增量式编码器
  - 正弦余弦传感器
  - 测速发电机

- 无传感器控制

  转子位置和速度根据电机电流和电压估算，无需使用其他传感器

2.2.2 根据控制信号

- 正弦标量控制

- 适用于正弦 PMSM 或交流感应电机的经典控制技术
- 矢量控制（FOC）
  - 适用于正弦 PMSM 或交流感应电机的高级控制技术，具有高动态和精密驱动性能

- 换向控制
  - BLDC 电机换向控制
  - 步进电机步进控制

- 其他控制技术

2.3 电机应用特性要求

图 2 显示的是一个常规三相电机控制拓扑。

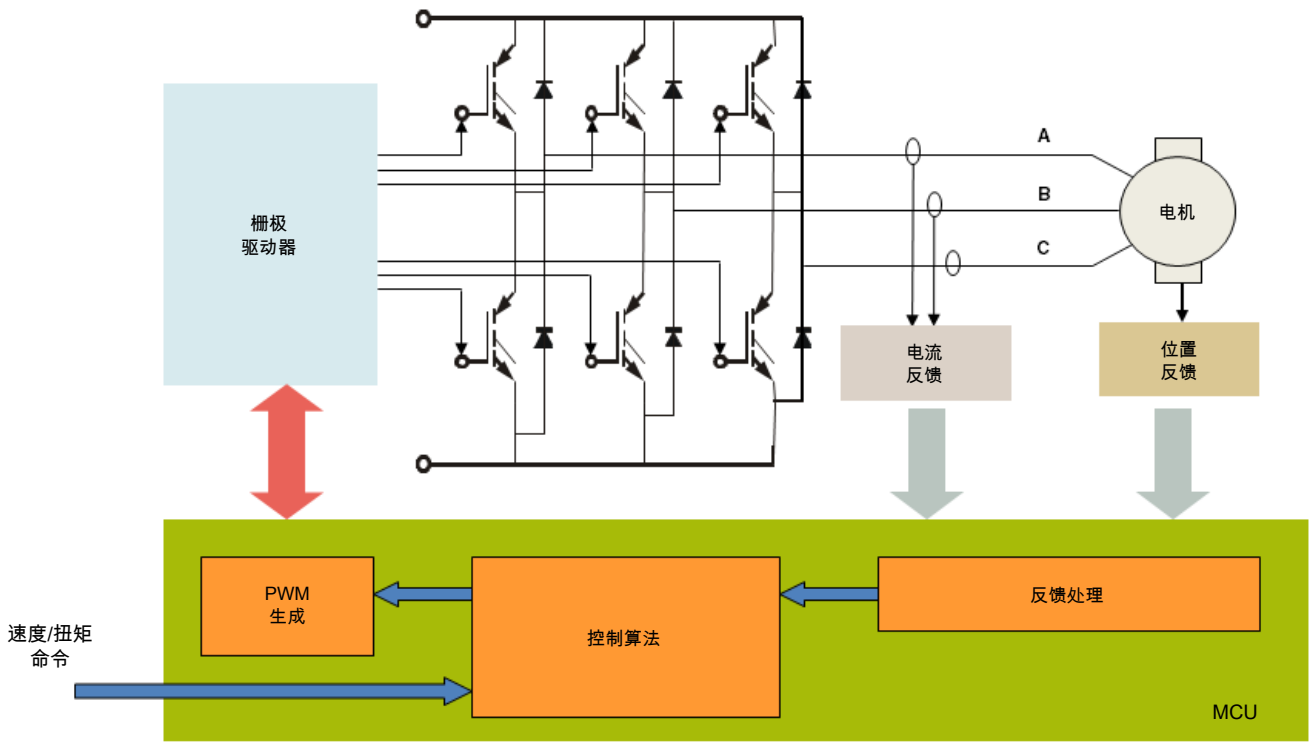

图 2. 常规电机控制拓扑

该拓扑基于三相功率平台，平台驱动器由 MCU 控制。MCU 输入包含电流、电压和位置反馈。这些信号由控制软件进行处理。PWM 生成三相功率平台所需的信号。后续章节将详细介绍电机控制应用特性要求。

2.3.1 由执行时间决定的电机控制过程

电机控制过程由同步任务和异步任务组成。这些事件由物理状态决定。事件之间的时间间隔取决于系统时间常数。

- 电机电气时间常数（绕组）通常为几十微秒
- 机械（转子机械惯性）

由物理状态决定的异步事件必须在几微秒至几十微秒内被处理器响应并执行。

- 低中断延迟
- 高优先级中断

2.3.2 由算法复杂度决定的电机控制过程

电机控制过程讨论了基于控制应用类型的各种算法复杂度。例如:

- 简单读取 -> 修改 -> 写入端口
  - BLDC 电机根据霍尔传感器换向

- 低复杂度算法，比如 PI 速度控制器
- 中等复杂度算法，比如后端
- 用于无传感器控制的 EMF 观察器——复杂算法，比如无传感器非线性交流感应电机控制

详情请参见 MQX 下的电机控制应用和实现 中的表格。

2.4 典型电机控制示例

电机控制算法的类型有很多种，此处重点介绍两种典型算法:

- BLDC 电机控制——表示异步换向事件的换向类型
- PMSM 电机控制——表示周期性执行不受电机速度影响的控制任务的高级控制技术

2.4.1 低复杂度应用示例——带霍尔传感器的 BLDC 电机控制

一种典型的低复杂度应用是带霍尔传感器的 BLDC 电机控制。BLDC 电机采用永磁体转子。电机的旋转靠定子磁通矢量的 6 步换向来实现。这一操作由三相电压系统提供，如图 3 所示。换向周期取决于转子速度，其持续时间可低至 200 µs。例如，运转速度为 10 000 rpm 的四极点三相 BLDC 电机以 500 µs 时间周期进行换向。通过霍尔传感器来同步换向时间。采用脉宽调制（PWM）技术控制电压幅度。

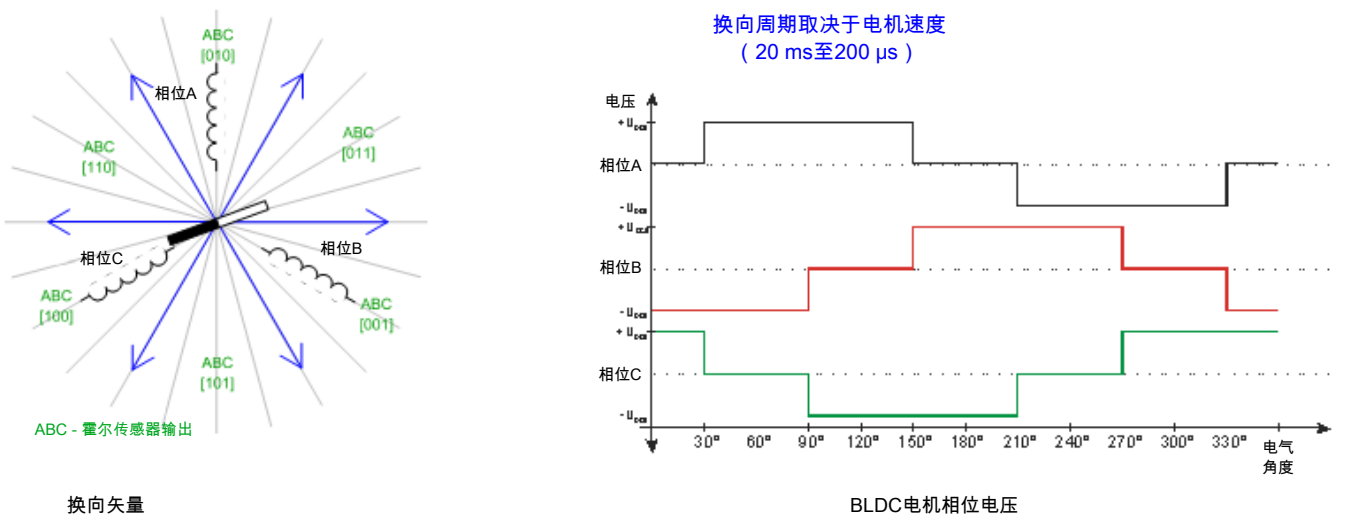

图 3. BLDC 电机换向

BLDC 电机控制系统请参见图 4 。霍尔传感器由解码器读取。这个过程由换向控制模块完成，实现 PWM 六步换向。该模块还提供换向周期，可实现速度控制。所需的 BLDC 矢量发送至 PWM 模块。电机速度和转矩限幅由控制器控制。所需的 PWM 占空比发送至 PWM 模块。PWM 模块根据所需的 BLDC 矢量和 PWM 占空比设置所需的相位信号。电机直流母线电流需要与 PWM 信号同步采样，这样 PWM 模块才能连接至 AD 转换器。采样直流母线电流供转矩限幅使用。

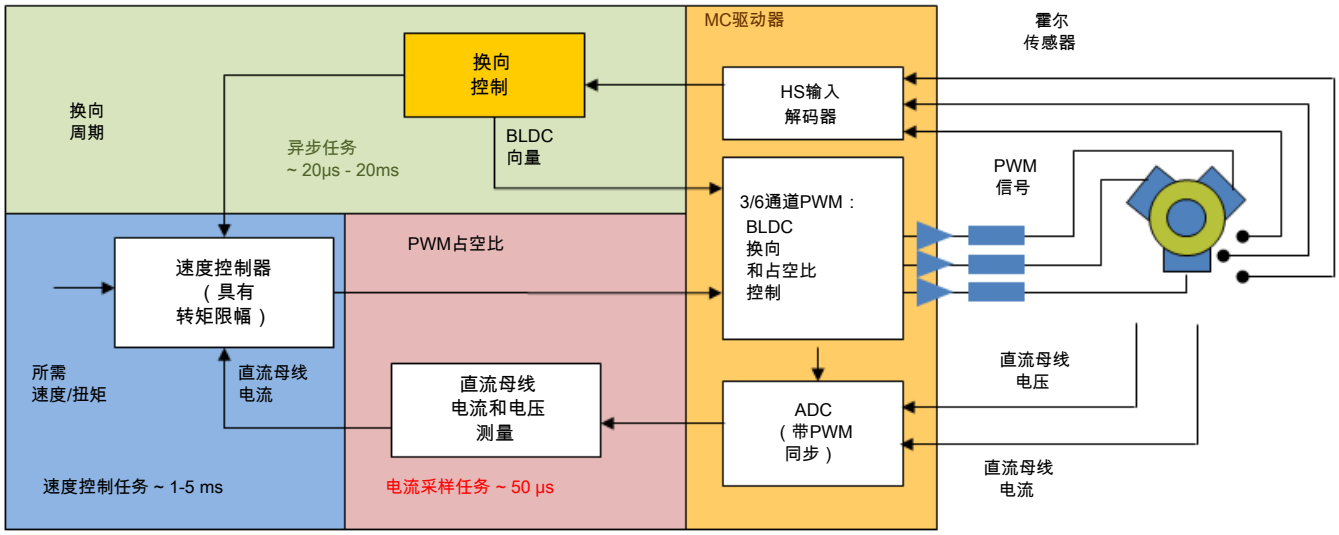

图 4. BLDC 电机控制

霍尔传感器 BLDC 速度控制应用基于三项任务:

- ADC — 电流（和电压）采样 — 周期（恒定周期，通常为 50 µs）
- 换向 — 与霍尔传感器变更事件同步（可变周期，持续时间为 20 ms 至 200 µs，具体取决于电机和速度）
- 速度控制 — 周期（恒定周期，通常为 1 ms 至 5 ms 周期）

上述 BLDC 电机控制应用中，最关键的时序是换向和电流采样。速度控制环路后台运行，所需的最高执行频率的任务为电流采样。然而，该任务的执行时间通常较短（通常不使用电流控制器）。霍尔传感器事件的换向调用具有可变频率。其执行时间（复杂度）取决于控制技术以及 PWM 模块的硬件支持，但由于复杂度低，因而执行时间通常较短。必须在较短的响应时间内处理完换向任务。部分飞思卡尔器件提供硬件支持，能够最大程度缩短延迟时间和简化控制软件。

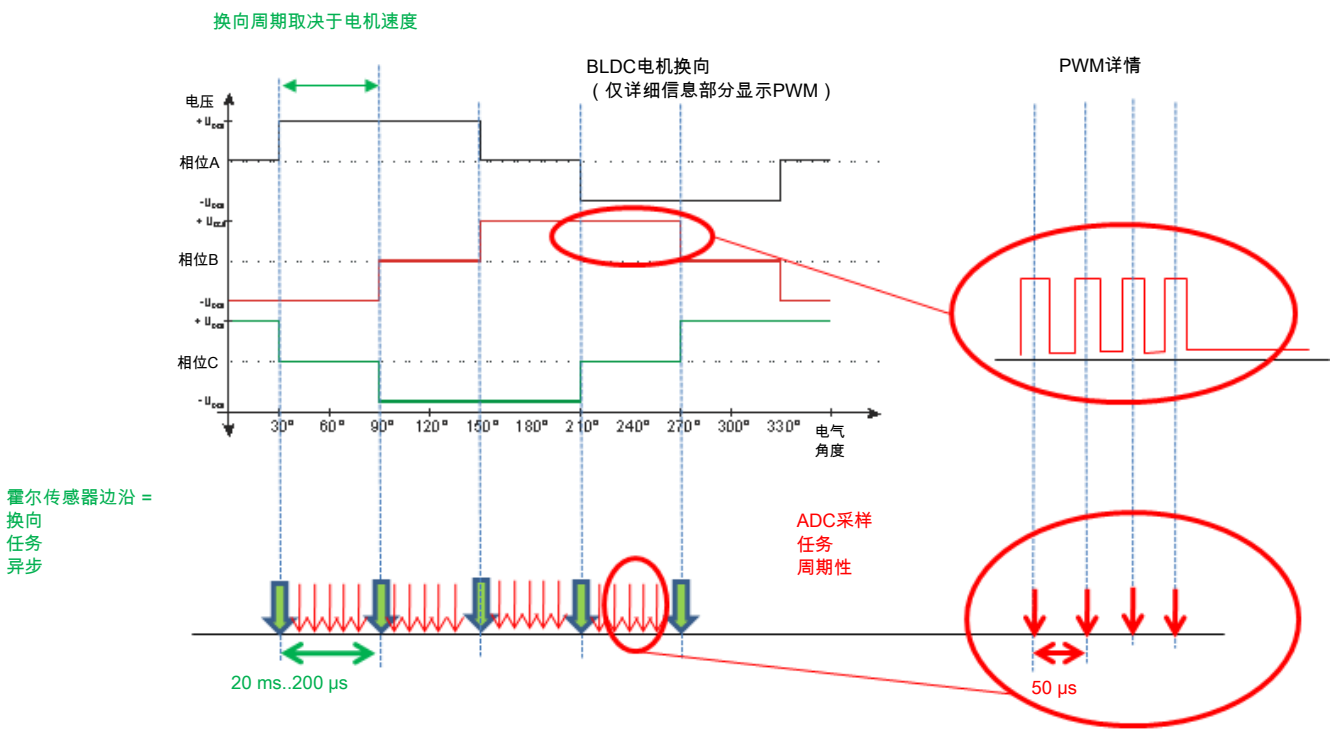

图 5. BLDC 电机控制 ADC 的采样和换向时序

采用低执行频率调用速度控制器任务。该调用频率恒定。此恒定调用周期通常为 1 ms 至 10 ms，具体取决于电机类型。

2.4.2 高复杂度应用示例 矢量控制

矢量控制是典型的高复杂度电机控制应用。其用于正弦电机（交流感应电机或永磁正弦电机）。一个典型的矢量控制应用是具有内部电流环路的速度控制。请参见图 6 。

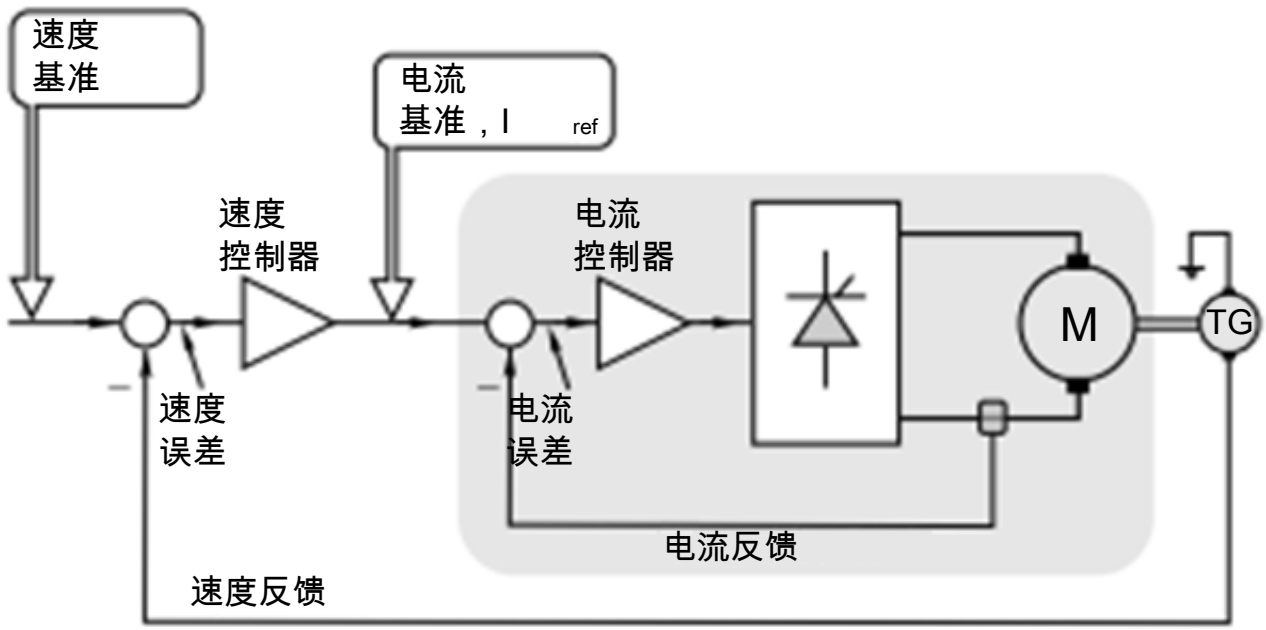

图 6. 具有内部电流环路的速度控制

速度矢量控制应用的应用结构详情请参见图 7 。电机采用正弦 PWM 电压供电。它具备两个软件控制环路：

- 慢速外部（速度）控制环路（1–5 ms）
- 快速内部（电流）控制环路（25–200 µs）

慢速外部（速度）控制环路和内部快速（电流）控制环路。执行时间方面最重要的部分是电流控制环路，其调用周期一般为 25 至 100 µs。该软件控制环路可将三相电流系统变换为二维静子相对坐标系 α β 。内部环路还可评估转子磁通位置，并通过 d-q 坐标将电流变换为转子磁通相对系统。d 轴和 q 轴电流由电流控制器进行控制。去耦之后，将其变换回三相 PWM 信号。因此，内部环路是一种中等至高复杂度的算法，调用周期较短。而带有速度检测装置和控制器的外部环路简单得多，但调用周期较长。外部控制环路对实时性要求没有内部控制环路高。

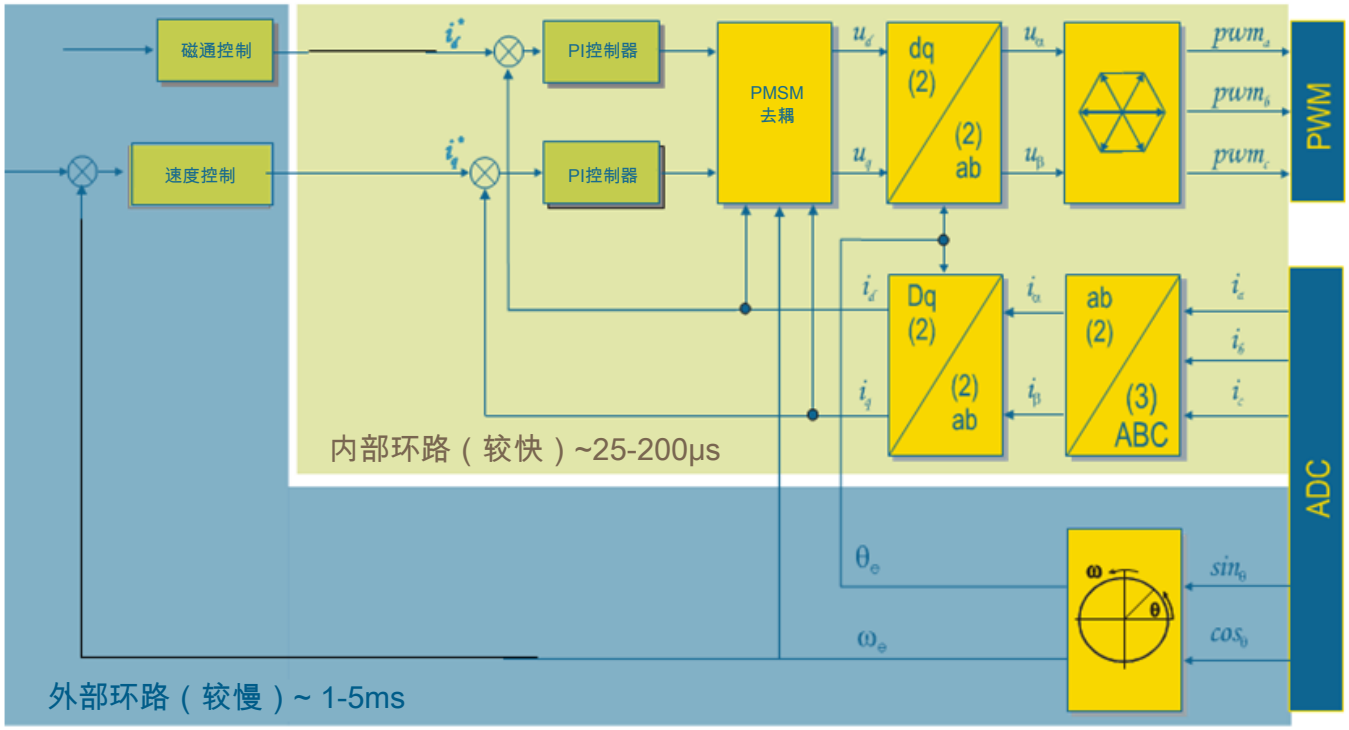

图 7. 复杂电机控制 — 矢量控制

12 kHz PWM 的快速环路时序请参见图 8 。每一个 PWM 周期内都可处理一次所需电流，通常在 ADC 中断子程序中处理。时序的主要部分是快速控制环路的执行。其余 CPU 时间用于较慢的控制环路和其他任务。

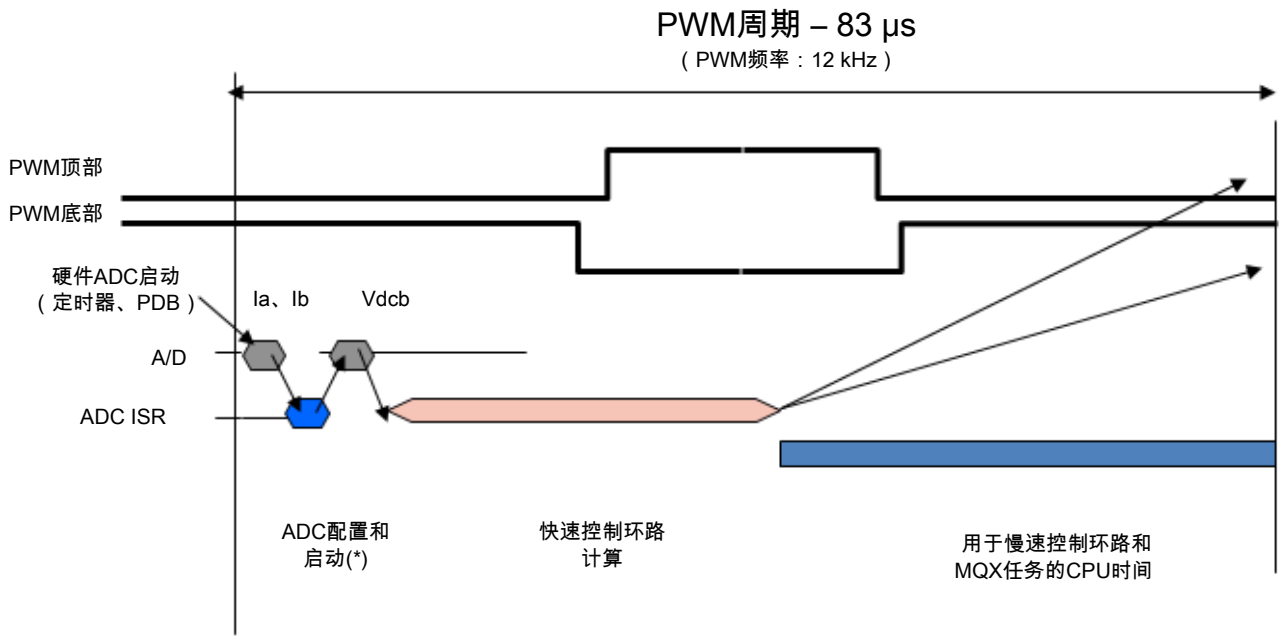

图 8. 矢量控制 – 快速环路时序

3 在 MQX 中集成电机控制

本节将说明如何在 MQX 中集成电机控制应用。

3.1 在 MQX 下进行电机控制的典型系统应用示例

图 9 是在 MQX 下执行电机控制应用的典型示例。

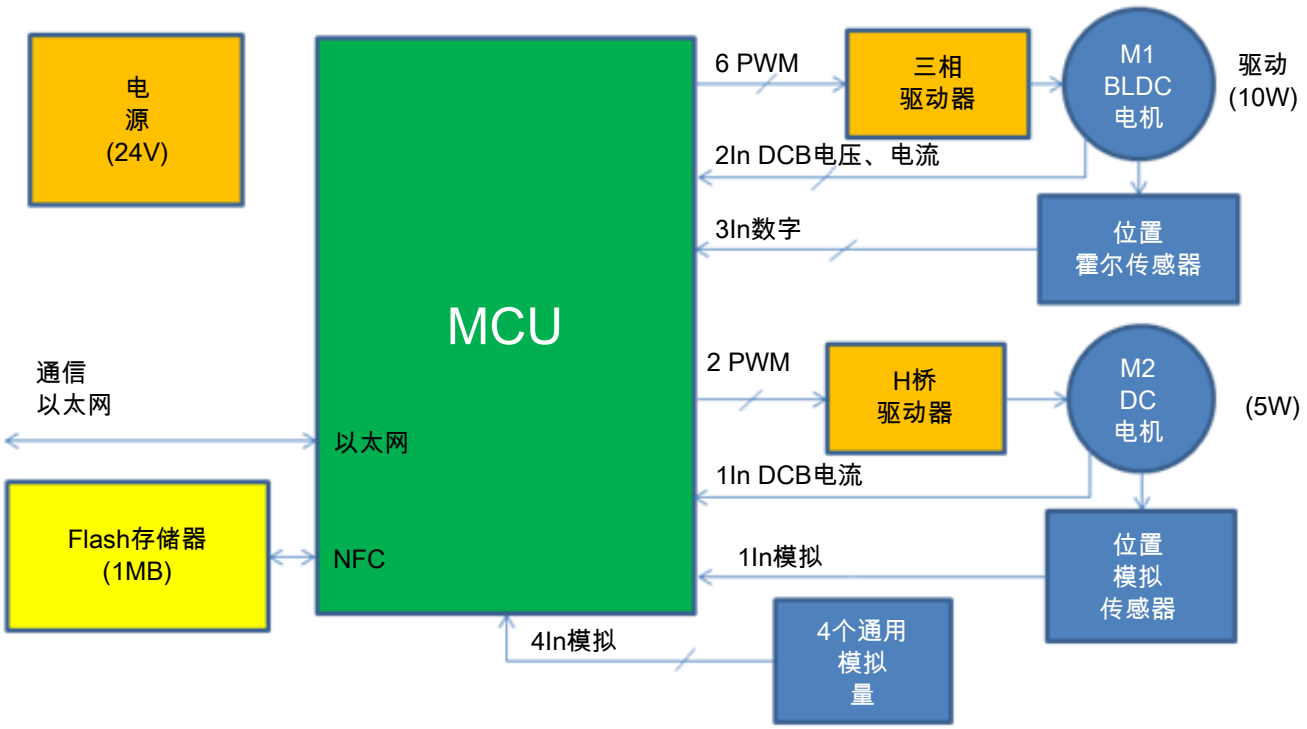

图 9. 带两个电机和模拟测量值的 Web 控制应用

相较于 CPU 上运行的纯电机控制应用而言，电机控制通常属于应用功能的一种。本应用控制一个具有专用传感器的 BLDC 电机和一个直流电机。其他应用功能通过以太网和模拟量采样与处理进行通信。MQX 系统支持以太网任务仲裁和其他功能。整个应用在单个 MCU 上运行。

3.2 何时在 MQX 下使用电机控制应用

MQX 并不是电机控制应用的典型操作系统。MQX OS 适合具有诸如 web 控制、USB 和 SDHC 卡读取等高级功能的大型应用，这些高级功能在专用设备（通常是单核）上运行。MQX 的主要优势在于，它包含用于 RTCS、以太网、USB 通信、MFS 文件系统以及许多其他应用的库。

在时间调度方面，高级电机控制应用本身基于常数采样（比如 ACIM 和 PMSM 正弦电机控制）或异步事件（比如 BLDC 电机换向控制），要求具备快速系统响应能力。大部分关键型事件所要求的响应时间通常为几微秒至几十微秒。

MQX 是一个复杂的系统，具有动态分配和 POSIX 调度功能。其系统默认节拍持续时间为 5 ms。这对大部分应用而言非常理想。然而，这同时也意味着，与电机控制要求相比，MQX 任务时间分辨率要长 1000 倍。因此，电机控制过程需要采用高优先级的中断来执行。这可通过标准 MQX 中断例程提供。从中断请求到服务程序执行之间的持续

时间通常为几微秒（具体取决于 CPU 版本和时钟速度）。如有需要，可用内核中断来执行电机控制算法。内核中断是自发性 CPU 中断，没有无谓的 MQX 消耗，并且执行持续时间极短。内核中断的劣势是不支持诸如事件或信号量等 MQX 功能。

3.3 MQX 下的电机控制 – 两种方法

有两种方法可以在 MQX 下写入电机控制软件：

- 已经存在专用的电机控制器的情况下
  电机控制过程由一个或多个内核中断（不支持诸如事件等 MQX 功能）或 MQX 最高优先级中断任务提供。因此，电机控制过程（任务）软件与标准非操作系统方法类似。但是，软件可以（虽然不一定）使用某些 MQX 器件外设驱动程序（包含在 MQX 安装程序中）。
  若采用这种方法，则 MQX 系统用于：
  - 所有任务的初始化，包括电机控制
  - 与电机控制无关的任务

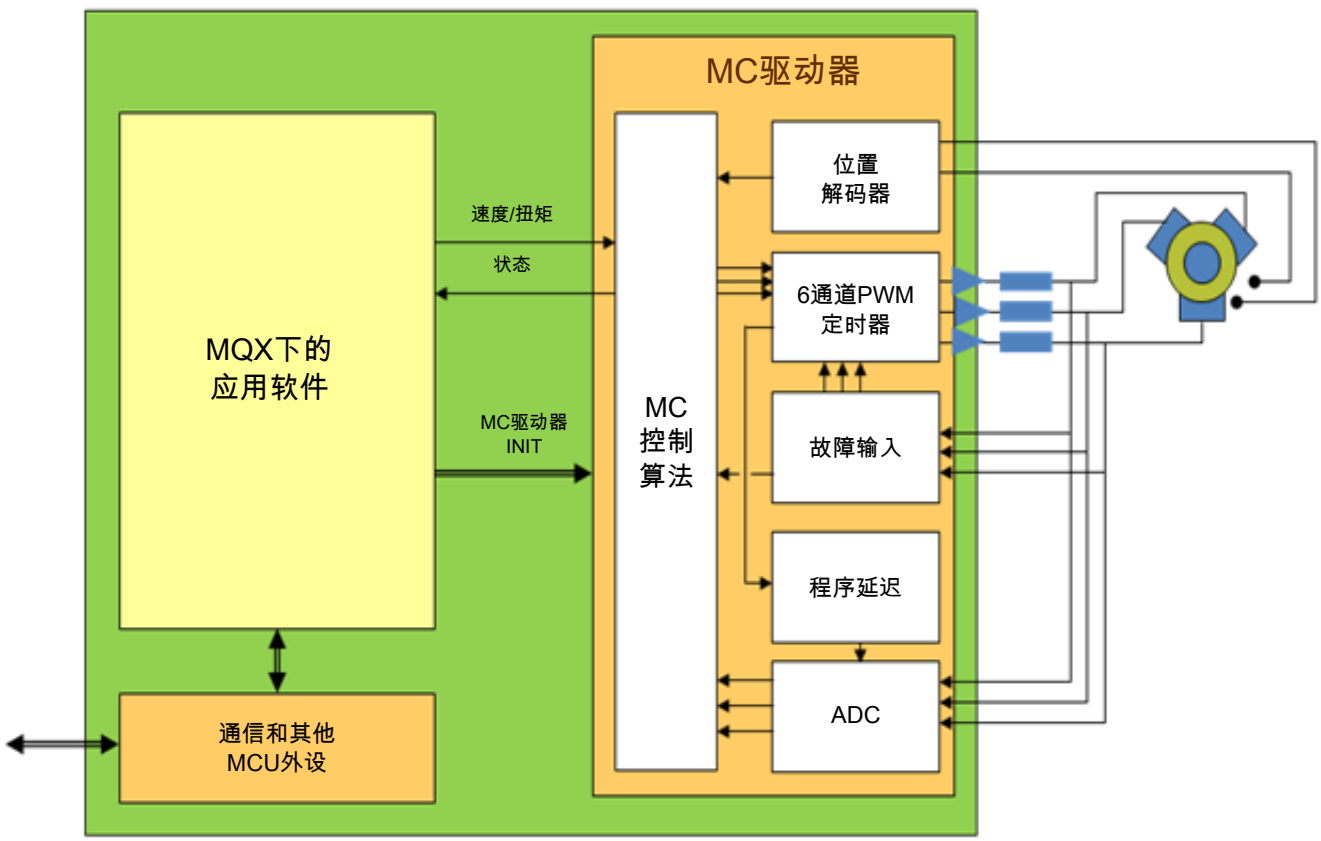

图 10. 专用电机控制驱动器

- 完全使用 **MQX** 完成电机控制
  电机控制应用程序通常由（多个）MQX 任务和中断提供。该软件一般使用部分 MQX 或客户写入的器件外设驱动程序。中断服务时间关键事件（例如换向、PWM 更新、位置传感器服务）和 MQX 任务执行非关键任务，如此，MQX 节拍周期延迟不重要（可能是速度控制环路）。

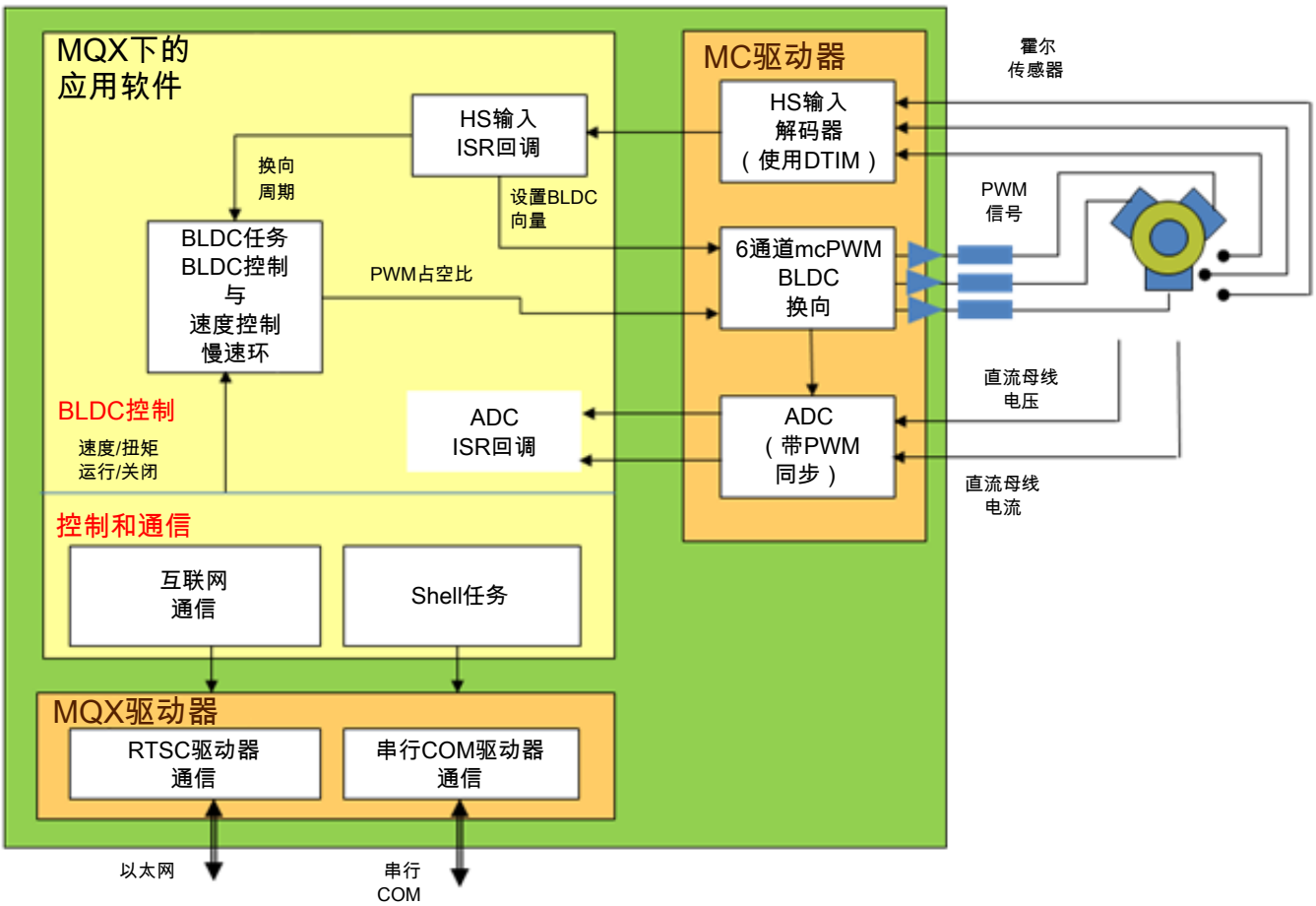

图 11. MQX 方法

3.4 专用电机控制驱动器示例

带专用电机控制驱动器和其他应用功能的 MQX 应用示例如图 12 所示。本应用示例是 PMSM 矢量控制。

应用特性包括:

- PMSM 矢量控制
- 以太网通信
- USB 接口
- 其他 MQX 任务

功能框图请参见图 7 。基于以太网的 CGI 接口、USB 控制任务在 MQX 下运行。应用控制 Shell 任务由 MQX 任务 1 提供。电机控制的上层软件层由 MQX 任务 2 实现。电机控制过程驱动器由一组 ADC 和定时器外设回调函数组成。中断回调可初始化为不受 MQX 系统影响的内核中断。然而，标准 MQX 中断可用于 MQX 中断处理的额外延时（通常为 2 µs，具体取决于 CPU 速度）并不重要的应用中。

标准 MQX 中断的优势在于，MQX 事件和标志可用作其他 MQX 处理过程的 API。MQX 系统管理所有 MCU 寄存器和中断标志备份。

专用电机控制驱动器优势:

- 可完全控制电机控制事件时序——MQX 仲裁不会导致延迟
- 仅可通过 MQX 或内核中断子程序来使时间关键型电机控制任务工作

劣势

- 软件开发更为复杂（未使用 MQX 仲裁）
- 外设完全分配给驱动器

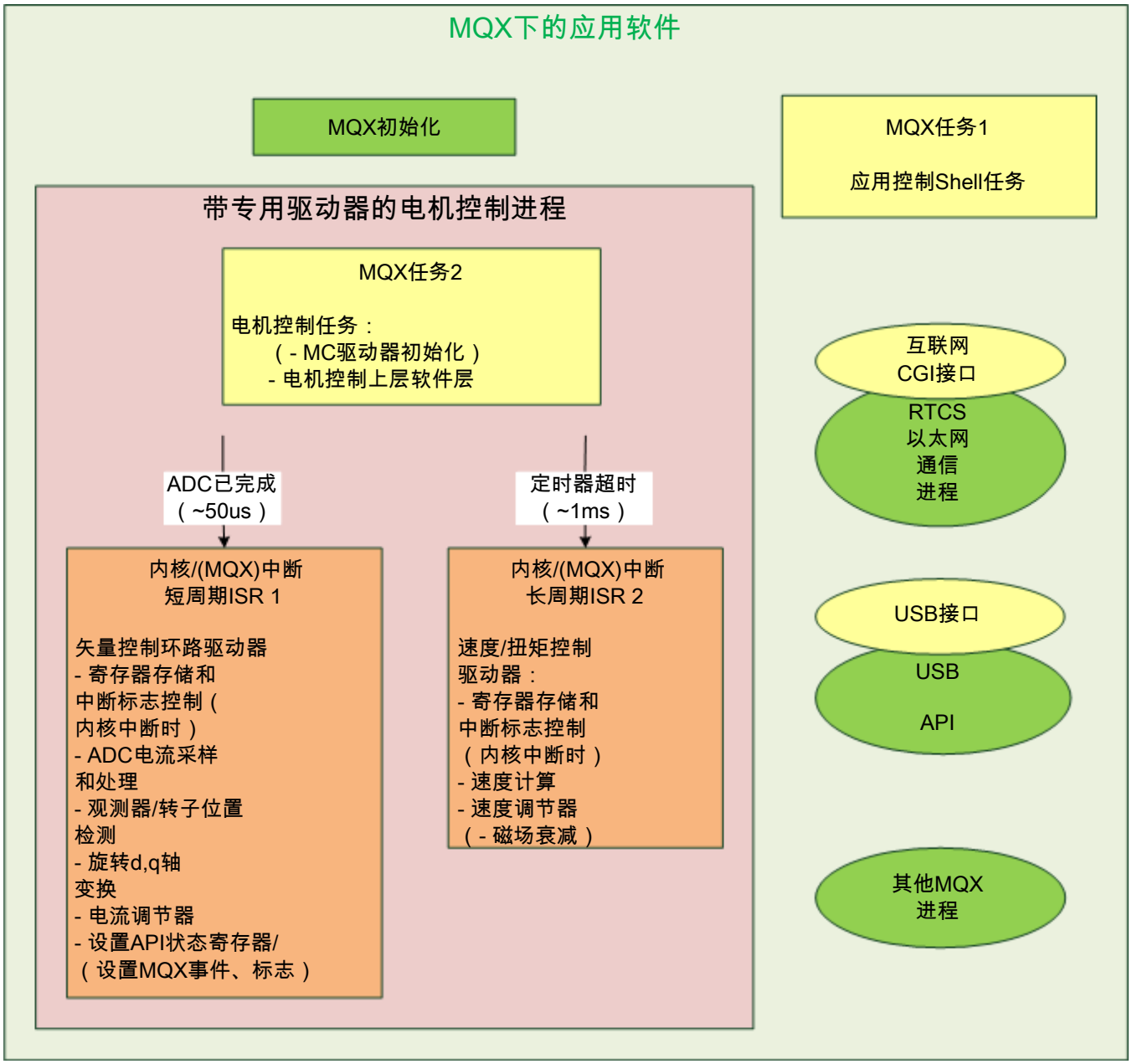

图 12. 电机控制过程在带专用驱动器的 MQX 下执行

3.5 MQX 方法示例

有关在 MQX 下执行电机控制应用且采用 MQX 方法的示例，请参见图 13 。此应用示例为带霍尔传感器的 BLDC 电机控制。

应用特性包括:

- 带霍尔传感器的 BLDC 电机控制
- 以太网通信

- USB 接口
- 其他 MQX 任务

功能框图请参见图 4。基于以太网的 CGI 接口、USB 控制任务和其他任务在 MQX 下运行。应用控制 shell 任务由 MQX 任务 1 提供。

通过采用 MQX 方法并使用 MQX 任务 2 和任务 3 以及 ADC 和霍尔传感器中断回调函数，可执行电机控制过程。中断回调在 MQX 任务 2 中初始化为 MQX 系统中断。标准 MQX 中断的优势在于，MQX 事件和标志可用作其他 MQX 任务的 API。MQX 系统处理所有 MCU 寄存器和中断标志。

专用电机控制驱动器优势:

- 软件开发较为简单（使用 MQX 任务仲裁、事件和信号量）

劣势

- 由于 MQX 限制（5 ms 节拍），时间关键型电机控制任务无法以这种方式执行。必须使用专用中断
- MQX 下的电机控制功能时间延迟

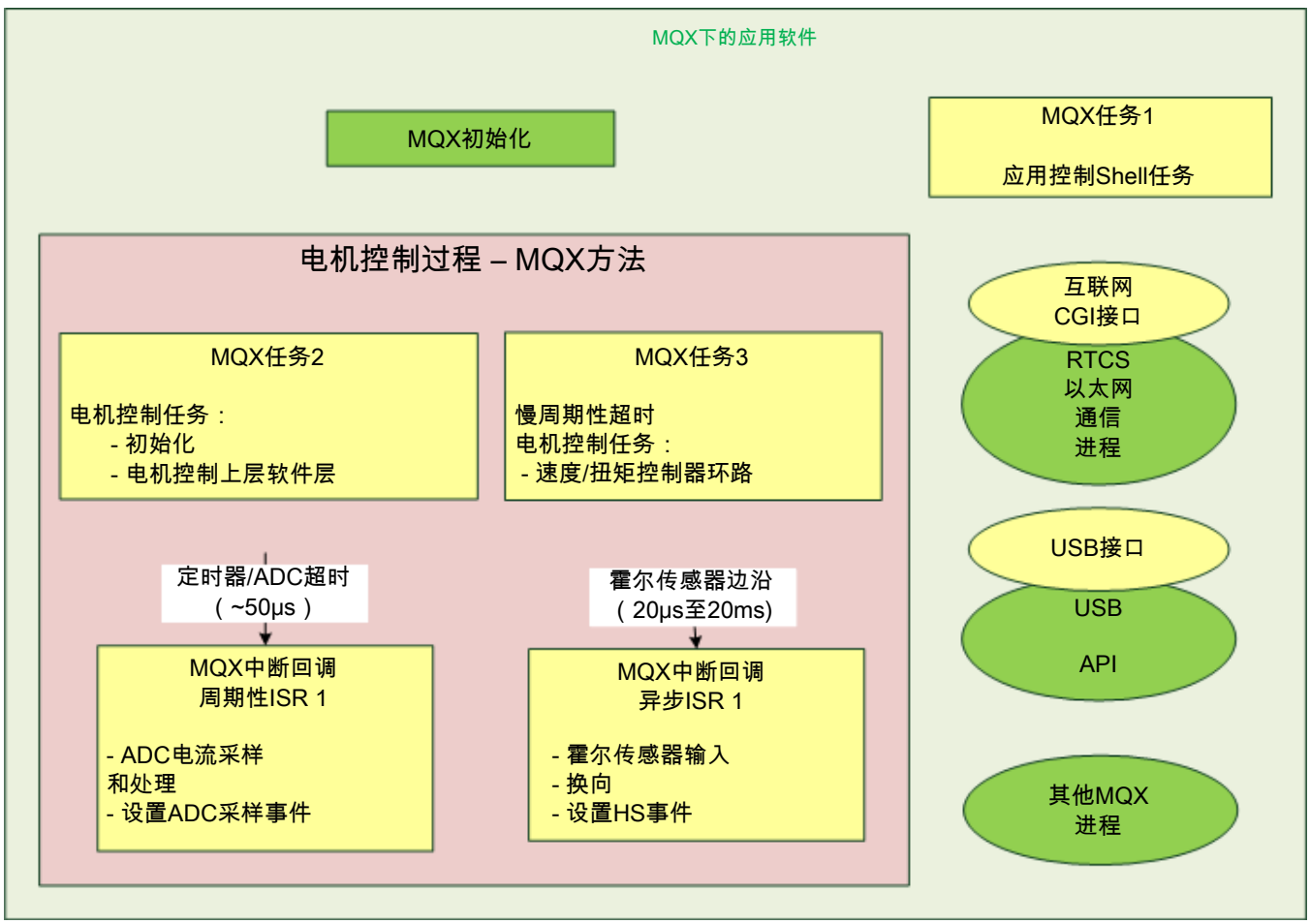

图 13. MQX 下的电机控制过程 – MQX 方法

3.6 MQX 下的电机控制应用和实现

MQX 下电机控制的实际实现取决于电机类型、控制算法和应用要求。下表显示了所需的任务和各种电机版本的电机控制过程在 MQX 下的实现。专用电机控制过程所需的任务取决于控制技术。所有电机控制技术均包含周期性任务与异步任务。下述表格显示了任务调用周期和典型任务复杂度。任务复杂度取决于具体的软件实现方式，因此表格中的数值是以 32 位 MCU 为时钟、频率为 50 MHz 时的持续时间示例。上述章节对函数调用进行了说明。最后一列显示 MQX 和 MQX 方法下的建议实现方式。

表 1. 直流电机

控制技术	采用此控制技术的主要任务	任务周期和同步事件示例	任务复杂度示例（50 MHz、32 位下的持续时间）	MQX 下的任务推荐部署	MQX 下的任务调用	MC 应用 MQX 方法
位置控制	周期性任务 1: — 位置控制器	周期性 1ms	低 （约 5 µs）	MQX 任务（或中断 MQX 回调） 控制器 采用 H 桥驱动器的算法	MQX 任务（/MQX 中断定时器回调）	MQX
	周期性任务 2:	周期性 50 µs	低 （约 3 µs）	用于 ADC 采样的中断调用驱动器	ADC 转换完成后的 MQX 中断回调	

表 2. BLDC 电机

控制技术	采用此控制技术的主要任务	任务周期和同步事件示例	任务复杂度示例（50 MHz、32 位下的持续时间）	MQX 下的任务推荐部署	MQX 下的任务调用	MC 应用 MQX 方法
使用霍尔传感器进行 BLDC 换向	周期性任务 1: — 直流母线电流和电压采样与处理	周期性 50 µs 与 PWM 同步 => ADC 转换完成	低 （约 2 µs）	用于 ADC 采样的中断调用驱动器，具有 PWM 同步，提供进一步处理所需的算法	ADC 转换完成后的 MQX 中断回调	MQX
	周期性任务 2: — 速度/扭矩控制器	周期性 1 ms	低 （约 5 µs）	采用 PWM 占空比驱动器的 MQX 任务（或中断 MQX 回调）控制器算法	MQX 任务（时间延迟）	
	异步任务 3: — 霍尔传感器状态读取和换向驱动器	异步 霍尔传感器变更事件时为 20 ms 至 200 µs	低 读/写 PORT 寄存器	用于定时器（霍尔传感器）输入和 PWM 换向驱动器的中断调用驱动器	定时器边沿捕捉时的 MQX 中断回调	
无传感器 BLDC 换向	周期性任务 1: — 直流母线电流和电压采样与处理	周期性 50 µs，与 PWM 同步 => ADC 转换完成	低 （约 2 µs）	用于 ADC 采样的中断调用驱动器，具有 PWM 同步，提供进一步处理所需的算法	ADC 转换完成后的 MQX 中断回调	无传感器 BLDC 控制驱动器和（子）任务驱动器（ADC 采样）
	周期性任务 2: — 速度/扭矩控制器	周期性 1 ms	低 （约 5 µs）	采用 PWM 占空比驱动器的 PI 控制器算法	定时器超时 MQX 中断回调	

下一页继续介绍此表...

表 2. BLDC 电机 (继续)

控制技术	采用此控制技术的主要任务	任务周期和同步事件示例	任务复杂度示例（50 MHz、32 位下的持续时间）	MQX 下的任务推荐部署	MQX 下的任务调用	MC 应用 MQX 方法
	异步任务 3: — 反电动势过零 — 换向时序	异步 — 反电动势过零事件时为 20 ms 至 200 µs	低至中等（25 µs）	中断调用无传感器 BLDC 驱动器 第 1 部分：反电动势过零和换向计算与定时器设置	MQX 中断回调比较器 -> 定时器输入捕捉	
	异步任务 4: BLDC 电机换向	异步 20 ms 至 200 µs	低 (10 µs)	中断调用无传感器 BLDC 驱动器第 2 部分：PWM 换向驱动器	定时器输出比较时的 MQX 中断回调	

表 3. PMSM 控制

控制技术	采用此控制技术的主要任务	任务周期和同步事件示例	任务复杂度示例（50 MHz、32 位下的持续时间）	MQX 下的任务推荐部署	MQX 下的任务调用	MC 应用 MQX 方法
带传感器的正弦 PMSM 控制	周期性任务 1: — 直流母线电流和电压采样 — 三相正弦生成	周期性 50 µs，与 PWM 同步 => ADC 转换完成	低 (10 µs)	用于 ADC 采样的中断调用驱动器，具有 PWM 同步 — 驱动器用于三相正弦生成	ADC 转换完成后的 MQX 中断回调	具有（子）任务驱动器的 MQX（ADC 采样、三相正弦生成）
	异步任务 2: — 转子位置识别。 — 三相正弦同步	传感器边沿处异步至少 100 µs	低 （约 5 µs）	用于定时器边沿输入捕捉的中断调用驱动器和用于三相正弦同步的算法	定时器边沿输入捕捉时的 MQX 中断回调	
	周期性任务 3: 速度/扭矩控制器	周期性 1m	低 （约 5 µs）	PID 控制器算法	定时器超时 MQX 中断回调	
采用传感器的 PMSM 矢量控制（编码器、正弦余弦等）	周期性任务 1: — 电流采样 — 位置识别 — 变换：a,b,c->α , β ->d,q, — 电流控制器	周期性 50 µs，与 PWM 同步 => ADC 转换完成	中等 (20 µs)	中断调用 PMSM 矢量控制驱动器	ADC 转换完成后的中断回调	PMSM 矢量控制驱动器和（子）任务驱动器
	周期性任务 2: 速度/扭矩控制器	周期性 1ms	低 （约 5 µs）	PI 控制器算法	定时器超时 MQX 中断回调	
无传感器 PMSM 矢量控制	周期性任务 1: — 电流采样 — 变换：a,b,c->α , β ->d,q, — 电流控制器， — 观测器计算	周期性 50 µs，与 PWM 同步 => ADC 转换完成	中等至高（40 µs）	中断调用无传感器 PMSM 矢量控制驱动器	PWM 同步或 ADC 转换完成后的中断回调	无传感器 PMSM 矢量控制驱动器和（子）任务驱动器

下一页继续介绍此表...

表 3. PMSM 控制 (继续)

控制技术	采用此控制技术的主要任务	任务周期和同步事件示例	任务复杂度示例（50 MHz、32 位下的持续时间）	MQX 下的任务推荐部署	MQX 下的任务调用	MC 应用 MQX 方法
	周期性任务 2: — 速度/扭矩控制器	周期性 1m	低 （约 5 µs）	PID 控制器算法	定时器超时 MQX 中断回调	

表 4. 交流感应电机控制

控制技术	实现此控制技术的主要任务	任务周期和同步事件示例	任务复杂度示例（50 MHz、32 位下的持续时间）	MQX 下的任务推荐部署	MQX 下的任务调用	MC 应用 MQX 方法
采用传感器的交流感应电机矢量控制（编码器、正弦余弦等）	周期性任务 1: — 电流采样 — 位置识别 — 变换：a,b,c->α, β ->d,q, 磁通估算器，— 电流控制器	周期性 50 µs，与 PWM 同步 => ADC 转换完成	中等 (35 µs)	中断调用 ACIM 矢量控制驱动器	ADC 转换完成后的中断回调	ACIM 矢量控制驱动器和（子）任务驱动器
	周期性任务 2: — 速度/扭矩控制器	周期性 1ms	低 （约 5 µs）	PID 控制器算法	定时器超时 MQX 中断回调	

4 演示 MQX 下的 BLDC 电机控制应用

为了在 MQX 下演示电机控制过程，我们设计了一款 BLDC 电机控制应用。

4.1 MQX 下的 BLDC 电机控制 – 应用特性

BLDC 采用飞思卡尔 TOWER 模块化硬件，集成三相功率平台板和 MCF5441x 控制器板。该应用通过霍尔传感器和以太网控制 24 V BLDC 电机。它在 MQX 下写入。以太网通信使用 MQX 库。BLDC 电机控制采用 MQX 方法提供。请参见图 14 。

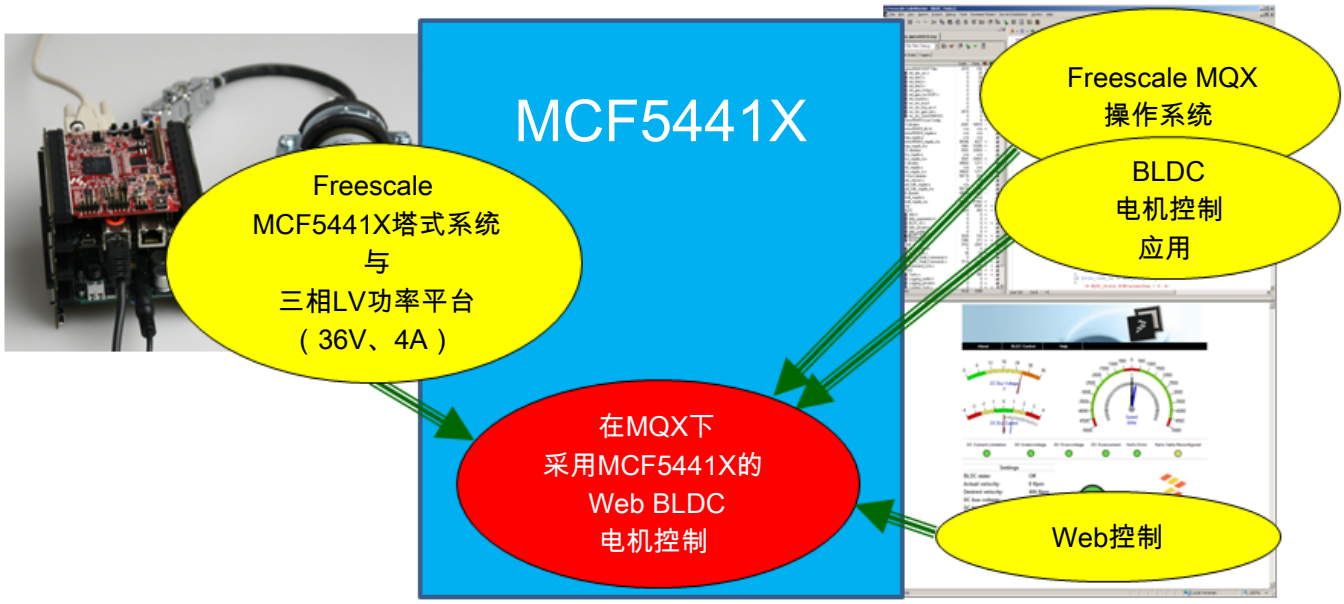

图 14. 演示 MQX 下的 BLDC 电机控制应用

5 MQX 下的 BLDC 电机控制 – 代码示例

这些是 MQX 下的 BLDC 电机控制 – 应用特性 所述应用的部分代码示例。

5.1 代码示例

该应用采用 MQX 方法，具有用户定义中断函数和 MQX 中断回调，以及其他与电机控制无关的应用任务。有关中断和 MQX_Task 参考，请参见图 14 。

这些示例的简化结构请参见 。该图未显示详细应用说明，而是显示了在下面所列代码中进行了评估的部分函数调用和数据结构。数据结构 `BLDC_State. Event` 是 MQX 事件的变量。`hsinput_dtim_info_ptr*`是 `hsinput_dtim_info_ptr` 所指向的结构，包含霍尔传感器输入数据，这些数据供 `hs_input_dtim_a_isr` 中断函数和 HS_input_callback 使用（参见下面的代码）。`MQX_template_list` 是 MQX 结构，可定义 MQX 任务（优先级、类别等）。详情请参见 MQX 文档的参考文献 第一项和第二项。

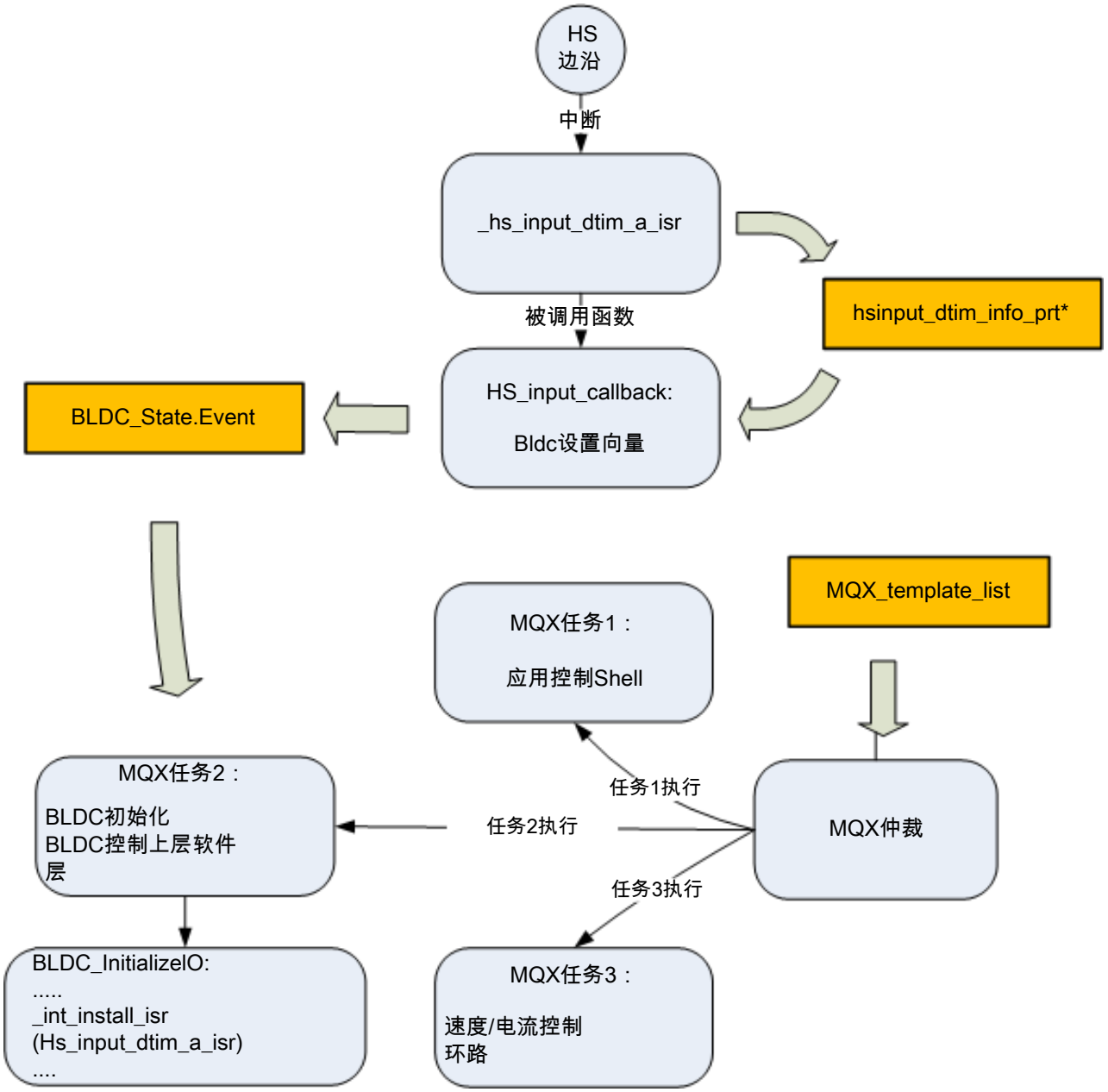

图 15. MQX 下的 BLDC 电机控制　　函数调用和数据结构示例

MQX 中断回调安装程序使用下面的 MQX 库函数提供应用初始化：

```
_int_install_isr。

……
      _int_install_isr(dtim_init_ptr_a->VECTOR,
                       (void (_CODE_PTR_)(pointer))_hsinput_dtim_a_isr,
                       (pointer) hsinput_dtim_info_ptr);
    …
```

霍尔传感器输入由 MCF5441x 处理器的 dtim 模块执行。用于霍尔传感器输入的 dtim 驱动器由_hsinput_dtim_a_isr 回调函数提供服务。该函数计算换向周期 `hsinput_dtim_info_ptr->LAST_EDGE_PERIOD`、读取霍尔传感器输入 `hsinput_dtim_info_ptr->HS_STATE`、清零外设挂起标志，并最终调用 HSInputCallbackIsr 算法：`uint_32 _hsinput_dtim_a_isr`。

```
(
      /* [IN] the address of the device specific information */
        MCF54XX_HSINPUT_DTIM_INFO_STRUCT _PTR_  hsinput_dtim_info_ptr,
    )
{
    /* calculation of the period between Hall Sensor Edges */
    hsinput_dtim_info_ptr->PREV_EDGE_TIME = hsinput_dtim_info_ptr->LAST_EDGE_TIME;
    hsinput_dtim_info_ptr->LAST_EDGE_TIME = hsinput_dtim_ptr->DTCR;
    hsinput_dtim_info_ptr->LAST_EDGE_PERIOD = hsinput_dtim_info_ptr->LAST_EDGE_TIME -
hsinput_dtim_info_ptr->PREV_EDGE_TIME;

     /* read & mask GPIO */
    hsinput_dtim_info_ptr->HS_STATE = (*(hsinput_dtim_info_ptr->GPIO_STR.GPIO_PPDSDR_PTR) & \
                                       hsinput_dtim_info_ptr->GPIO_STR.PIN_MASK)>> \
                                       hsinput_dtim_info_ptr->GPIO_STR.PIN_SHFT;
    /* clear pending flags */
    periphBitSet(MCF54XX_DTIM_DTER_CAP, &(hsinput_dtim_info_ptr->DTIM_A_PTR->DTER));

     /* call HS_input_callback  */
    if(hsinput_dtim_info_ptr->HSINP_CALLBACK_ISR != NULL)
    {
        hsinput_dtim_info_ptr->HSINP_CALLBACK_ISR (hsinput_dtim_info_ptr);
    }
    return( MQX_OK );
}
```

然后调用中断回调算法 HSInputCallbackIs，以根据霍尔传感器状态提供 PWM 换向。它还处理换向周期，并设置 MQX 事件 BLDC_HS_INPUT_CHANGED：

```
void HS_input_callback (pointer parameter)
{
    MCF54XX_HSINPUT_DTIM_INFO_STRUCT_PTR  hsinput_dtim_info_ptr = parameter;
    /* set required sector_u8 variable according to table and Hall sensor input */
    sector_u8 = bldc_3pps_hs_table[hsinput_dtim_info_ptr->HS_STATE];
    /* commutate the pwm sector according to , sector_u8 variable */
    _3ppspwm_bldc_set_vector_sector_6s_ss_compl(pspwm_info_ptr, sector_u8);
    /* read Hall sensor period */
    BLDC_State.ActualPeriodSample = _hsinput_read_period(hsinput_info_ptr);
    /* set BLDC_HS_INPUT_CHANGED  event */
    _lwevent_set(&BLDC_State.Event, BLDC_HS_INPUT_CHANGED);
}
```

任务 2 是 MQX 任务函数的一个示例，该例提供联网初始化、BLDC 电机控制应用初始化和电机控制上层软件层（参见图 13 任务 2）：

```
void Task2 (uint_32 data)
{
    /* Initialize ethernet networking */
    initialize_networking();
    /* Initialize operating parameters to default values */
    BLDC_InitializeParameters();
   /* create light weight event structure */
    _lwevent_create(&BLDC_State.Event, 0);
    /* Configure and reset outputs */
    BLDC_InitializeIO();
    while(TRUE)
    {
        /* Motor Control Upper S/W Layer */
        …
    }
```

任务 1 函数仲裁由 MQX 提供。任务列表在 MQX 中以标准方式定义：

```
const TASK_TEMPLATE_STRUCT MQX_template_list[] =
{
    /* Task Index,                      Function,         Stack,  Priority,  Name,
Attributes,                        Param,  Time Slice */
    {      TASK1,                        MQX_Task2      2000,        3,    "BLDC",
MQX_AUTO_START_TASK,    0,      0           },
```

```
    { BLDC_CONTROL_TASK, MQX_Task3,  1500,          4,      "BLDCCtrl",
0,                              0,      0          },
     { SHELL_TASK,                MQX_Task1,     2900,        12,   "Shell",
MQX_AUTO_START_TASK,      0,      0           },

{ ..........,                                                  ..,            .,          ..,
      ......,                                                  ....,        .,       ..
},
    {0}
};
```

6 参考文献

本节包含与本应用笔记相关的在线资源列表。

1. 参考手册：*Freescale FSLMQX™ RTOS*（文档 FSLMQXRM）
2. 用户指南：*Freescale FSLMQXTM* 实时操作系统（文档 FSLMQXUG）
3. 设计参考手册：使用 *MC9S08MP16* 的三相无传感器 *BLDC* 电机控制（文档 DRM117）Freescale Semiconductor，2009
4. 设计参考手册：使用 *MC56F8013*、将滑动模式观察器用作压缩器的无传感器 *PMSM* 矢量控制（文档 DRM099）Freescale Semiconductor，2008
5. 使用 *DSP56F80x* 的三相 *PM* 同步电机矢量控制，作者：Prokop L.、Grasblum P.，AN1931，Motorola，2002

7 定义和首字母缩略词

本节列出本应用笔记中涉及的多个首字母缩略词。

AN — 应用笔记

ACIM — 交流感应电机

API — 应用接口

BLDC — 无刷直流电机

FOC — 磁场定向控制

Freescale MQX™ — 由飞思卡尔半导体提供的实时操作系统 MQX

在本应用笔记中，电机控制是指控制 BLDC PMSM、交流感应电机或其他电机的过程。

MQX™ — 实时操作系统

Tick — 操作系统时间单位（也代表最小时间分辨率）

OS — 操作系统

POSIX — 便携式操作系统接口，由 IEEE 制造，并由 ANSI 和 ISO 进行标准化。MQX 符合 POSIX.4（实时扩展）和 POSIX.4a（线程扩展）标准。

PMSM — 永磁同步电机

RTOS — 实时操作系统

RTCS — 嵌入式互联网协议栈为 MQX 平台提供 IP 网络。RTCS 包含多种 TCP/IP 联网应用协议，并使用 MQX RTOS 驱动程序实现以太网或串行连接。

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number AN4254
Revision 0, 5/2011