Freescale Semiconductor, Inc.
应用笔记

文档编号：AN4935
修订版0，2014年5月

利用 Kinetis KV10 实现适用于风扇的 PMSM 无传感器磁场定向控制 (FOC)

1 简介

本应用笔记是“PMSM 无传感器磁场定向控制设计参考手册”(DRM148) 的补充。对应用笔记的内容作了增强，具备了抗风功能，可以在风扇叶片因风的作用而受迫转动时处理特定的启动状况。本文提出的解决方案既不识别受迫转动的速度，也不识别其方向。旋转的转子缓缓停止后，便可检测到转子的初始位置且在转子不对齐情况下开始旋转。

本应用笔记介绍制动转子和检测转子初始位置所使用的算法、软件修改、该应用所用的 MCU 外设、硬件设置以及测量结果。

2 抗风功能

抗风功能是风扇应用必需的特性。大多数叶片较大的吊式风扇都很容易因气流或房间中其他风扇的运转所引起的空气运动而发生被动旋转。因为旋转的 PMS 电机可产生反电动势，要防止电流浪涌引起系统故障，则在施加电压矢量前让转子缓缓停止。

目录

制动转子的方法很简单。在制动开始时，所有三相的下桥臂晶体管打开在 10%。如果转子在转动，所产生的反电动势会激励一个小电流。该电流可通过分流电阻测量，并由 MCU 检测到。当该电流开始流动时，电机就会产生制动扭矩，使转速降低。转动中的转子其能量以热量的形式消耗，主要消耗在定子绕组中。如果电流高于预定义阈值（本应用中为标称值的 10%），则 MCU 要等到电流降低后才开始工作。如果电流降至阈值以下，PWM 占空比就会提高，直至电流再次达到阈值。这样，电机就被轻缓地制动到静止状态，下桥臂晶体管逐渐开启到 100%。图 1 所示的流程图描述了上述制动过程。

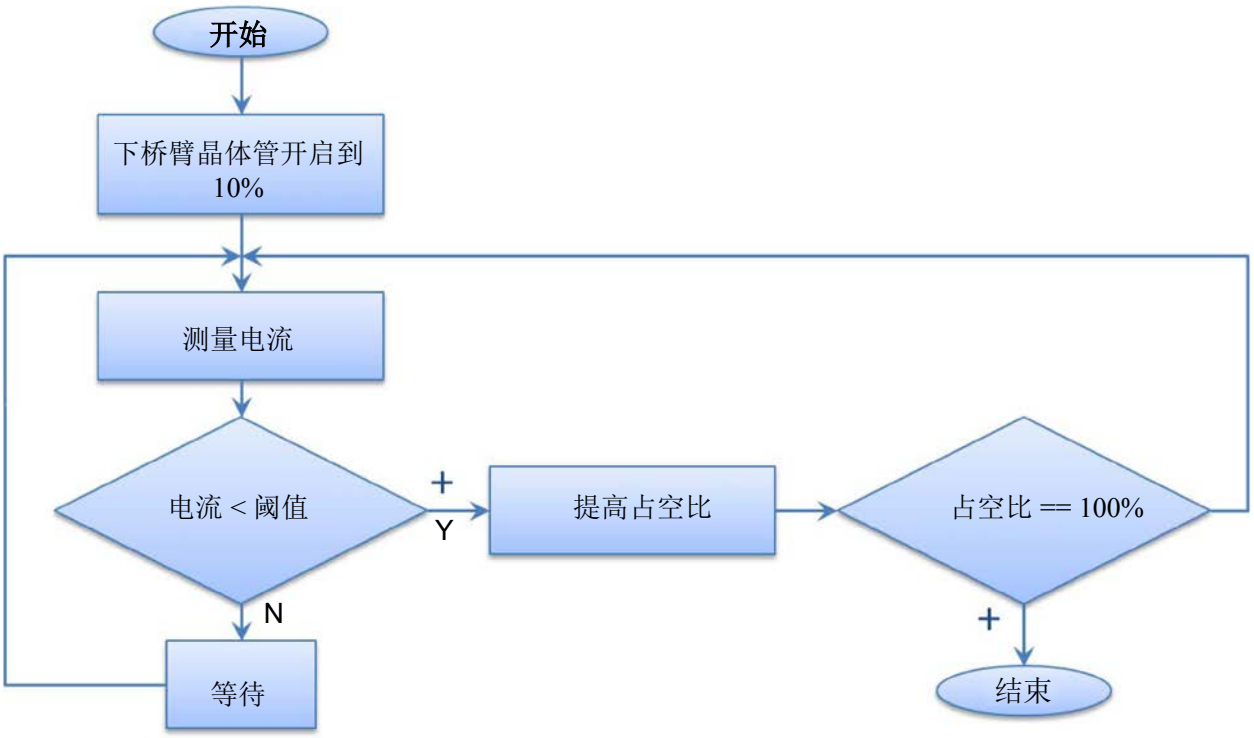

图 1. 制动过程

为了使该系统的可靠性可进一步提高。可采用一个定时器来限制制动过程，以在占空比为 100%时仍检测到相当大的电流为例，该系统可以发出一个故障标志，这样就会禁止电机启动。

2.1 转子初始位置检测

为了成功执行矢量控制算法，必须知道控制环的每一个计算时刻的转子位置。然而，当电机处于静止状态时，无法从反电动势接收位置信息。为了获得电机静止时的位置，常常采用所谓的“转子对齐”法。在对齐期间，沿 d,q 坐标系的 d 轴产生电压矢量，转子移动到转子与定子磁场对齐时的位置。在风扇应用中，由于吊式风扇叶片的惯性很大，因此使用该对齐法并不太合适。

转子停止后，执行初始位置检测算法。该检测方法基于永磁体引起的定子铁芯饱和效应。由于转子磁体导致饱和，定子电感会随着转子位置不同而变化。对定子绕组施加一个恒定电压时，通过绕组中的电流变化率可以观察到该效应。通过测量定子绕组中因为电感变化所引起的电流变化率，便可估算转子磁体和定子绕组之间的相对位置。“无位置传感器的 BLDC 电机的全新启动方法”一文中详细介绍了这种方法，参见第12章节“参考文献”。在本文中仅说明基本原理和软件实现。

估算转子位置时基于定子铁芯的非线性磁化特征。这是由于定子铁芯在靠近转子永磁体的磁极时会被强烈磁化。因此，如果定子绕组靠近转子的磁极，因定子铁芯的磁饱和定子绕组中沿磁化方向流动的电流变化率会大于沿相反方向流动的电流变化率。

本应用将 6 个具有相同长度和相同时间量的基本电压矢量（用于“空间矢量 PWM”模块）逐一施加于绕组。这样就可以产生 6 个对应的电流脉冲，然后根据这些电流脉冲的最大值之差，就可估算出分辨率为 30 电角度的转子位置。根据估算的转子初始位置可以产生适当的电流矢量，从而实现最大启动扭矩。图 3表示的是初始位置检测的简化过程。

“无位置传感器的 BLDC 电机的全新启动方法”一文（参见第12章节“参考文献”）提出了根据测得的直流母线电流估算转子初始位置的方法。由于在本应用中不测量此电流，因此根据相电流（在各扇区中，考虑另一相电流）来估算初始位置，如图 2所示。

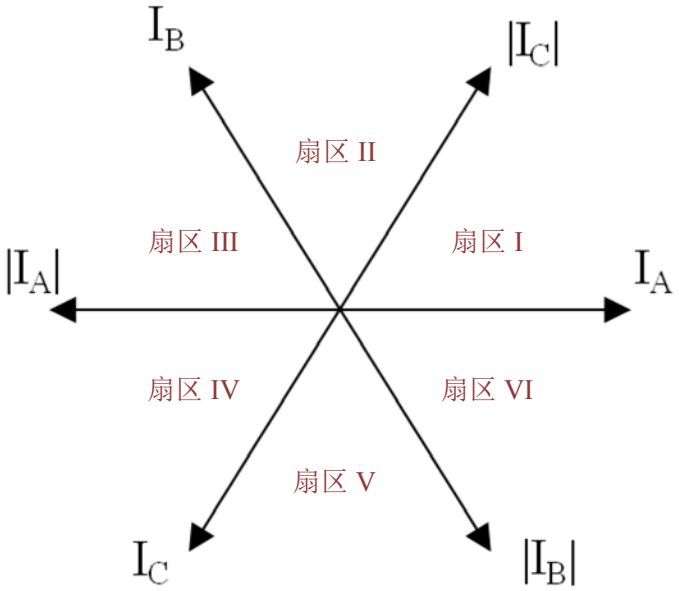

图 2.　初始位置检测—利用相电流在特定扇区中进行估算

要使抗风应用适用于任何电机，只需知道定子线圈的电气时间常数即可。由于存在 CPU 性能约束条件（即执行时间和快速控制环的 0.1 ms 周期），电机的最小电气时间常数必须是 0.1 ms。如果未针对特定电机正确调整应用，那么测得的电流之差将会很小，导致转子初始位置的检测不可靠。这种情况下会终止转子位置检测过程，并置位故障标志。随后，应用继续执行电压对齐。

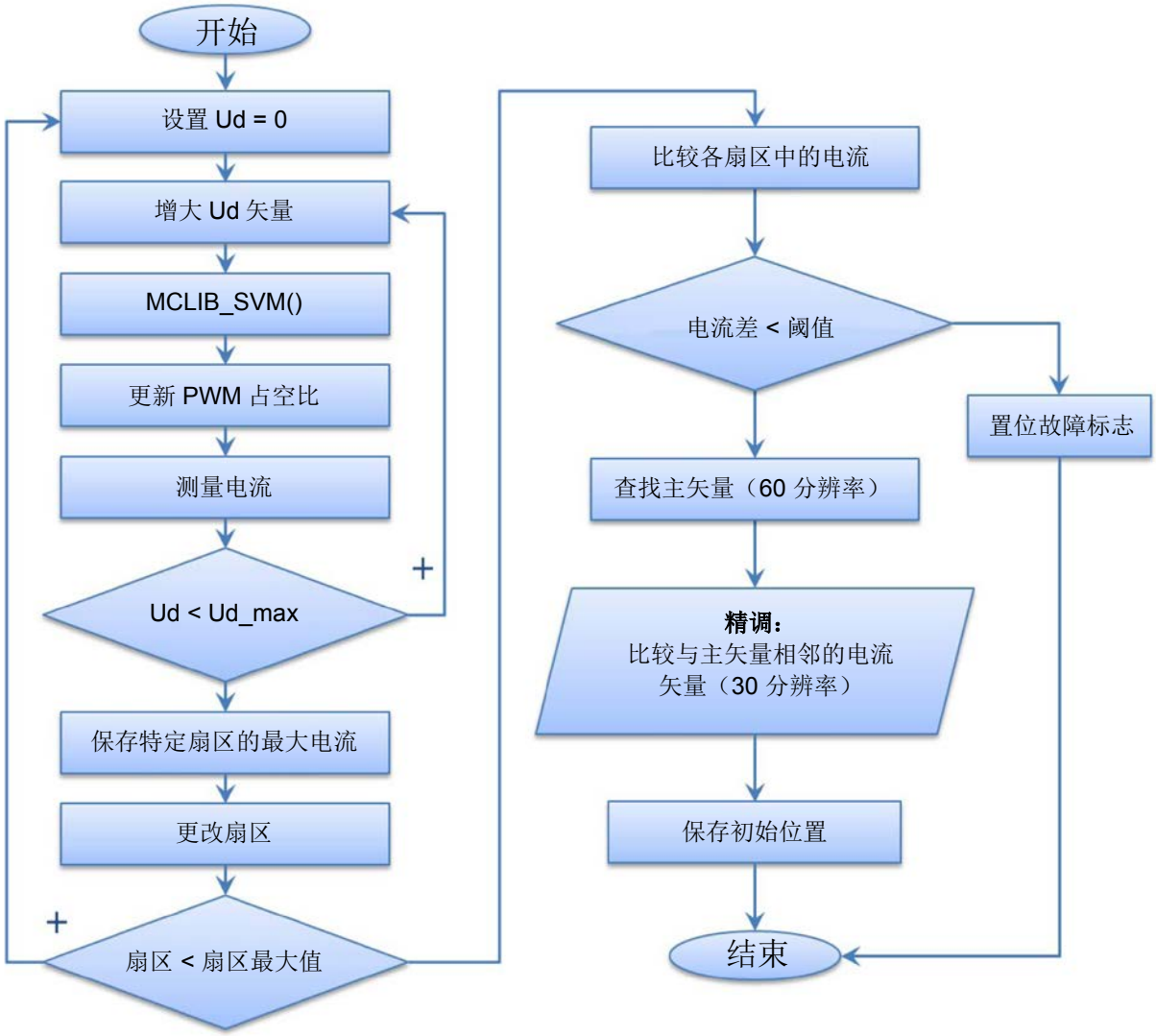

图 3. 初始位置检测过程

3 软件说明

3.1 应用状态机

应用状态机与 DRM148 所述的原始状态机在几个细节上有所不同。Run（运行）状态得到增强，包括两个附加子状态——*Brake*（制动）和 *PosDetection*（位置检测），以便支持抗风特性和未对齐情况下的转子初始位置检测。*Calib*（校准）在 Brake 子状态之后执行，这是因为用于测量电流的 ADC 通道的偏置校准是在转子处于静止状态时执行。应用启动时，电机正在旋转，无法测量电流偏置。因此，初始电流偏置值是在应用开发期间形成的。这对性能的影响很

小。电流偏置值几乎是原始 ADC 最大值的一半；也就是说，对于 12 位分辨率，它是 2^{11} = 2048。*Ready*（就绪）子状态也被修改。在原始应用中，*Ready* 子状态下的所有 PWM 输出都使能且设置为 50%。电机运转时，这可能引起电流浪涌，导致系统故障。图 3显示了 Run 状态及其子状态和转换条件。

当状态机处于 Run 状态时，会调用 Run 各种子状态。Run 子状态函数说明如下：

- **Ready（就绪）：**
  - 下桥臂晶体管开启 10%，上桥臂晶体管保持关断。
  - PWM 输出保持禁用。
  - 测量相电流和直流母线电压。
  - 设置 ADC 通道。
  - 初始化某些变量。
- **Brake（制动）：**
  - 使能 PWM 输出，若电机在旋转，则进行制动。
  - 测量相电流。
  - 下桥臂晶体管的占空比提高到 100%。
- **Calib（校准）：**
  - 占空比设置为 50%，上桥臂和下桥臂晶体管使能。
  - 测量电流通道失调并滤波。
- **PosDetect（位置检测）：**
  - 测量电流。
  - 执行位置检测算法。
  - 若转子初始位置检测失败，则应用进入 Align（对齐）子状态，否则进入 Startup（启动）子状态。
- **Align（对齐）：**
  - 测量电流。
  - 设置 ADC 通道。
  - 调用转子对齐算法。
  - 更新 PWM。
  - 测量直流母线电压。
  - 对齐时间之后，系统切换到 Startup（启动）子状态。
- **Startup（启动）：**
  - 测量电流。
  - 设置 ADC 通道。

— 调用反电动势观测器算法以估算转速和位置。
— 对估算的转速进行滤波。
— 调用 FOC 算法。
— 更新 PWM。
— 测量直流母线电压并滤波。
— 调用开环启动算法。

- **Spin（旋转）：**
  — 测量电流。
  — 设置 ADC 通道。
  — 调用反电动势观测器算法以估算转速和位置。
  — 对估算的转速进行滤波。
  — 调用 FOC 算法。
  — 更新 PWM。
  — 电机旋转。
  — 测量直流母线电压。
  — 调用速度斜率和速度 PI 控制器算法。
  — 评估速度命令。
- **Freewheel（惯性运行）：**
  — 禁用 PWM 输出，模数设置为 50%。
  — 测量电流。
  — 设置 ADC 通道。
  — 测量直流母线电压。
  — 系统在该子状态下等待一定的时间，该时间由电机惯性决定，也就是要等到电机停止为止。
  — 系统评估相关状态，然后进入 Align 或 Ready 子状态。

Run 子状态还具有子状态之间迁移的函数。子状态迁移函数说明如下：

- **Ready > Brake**—非零速度命令；进入 Brake 状态。
  — 上桥臂晶体管使能。
- **Brake > Calib**
  — 上桥臂和下桥臂晶体管使能。
- **Calib > PosDetect**
  — 初始化用于位置检测算法的变量。
- **PosDetect > Align**—位置检测失败；进入 Align 状态。
  — 初始化某些变量（电压、速度、位置）。

— 设置对齐时间。

- **PosDetect > Startup**—位置检测成功。
  - — 清除变量。
  - — 将检测到的转子位置传递给启动数组。
  - — 进入 Startup 状态。
- **Align > Ready**—零速度命令。
  - — 进入 Ready 状态。
- **Align > Startup**—对齐完成。
  - — 初始化滤波器和控制变量。
  - — 进入 Startup 状态。
- **Startup > Spin**—进入 Spin 状态。
- **Startup > Freewheel**—不发生任何操作。可用于处理启动故障状况以实现更可靠的应用。
- **Spin > Freewheel**—零速度命令。
  - — PWM 输出禁用。
  - — 进入 Freewheel 状态。
- **Freewheel > Ready**—惯性运行时间到期。
  - — PWM 输出使能。
  - — 进入 Ready 状态。
- **Freewheel** > **Brake**—非零速度命令。
  - — PWM 输出使能。
  - — 初始化某些变量（电压、速度、位置）。
  - — 进入 Brake 状态。

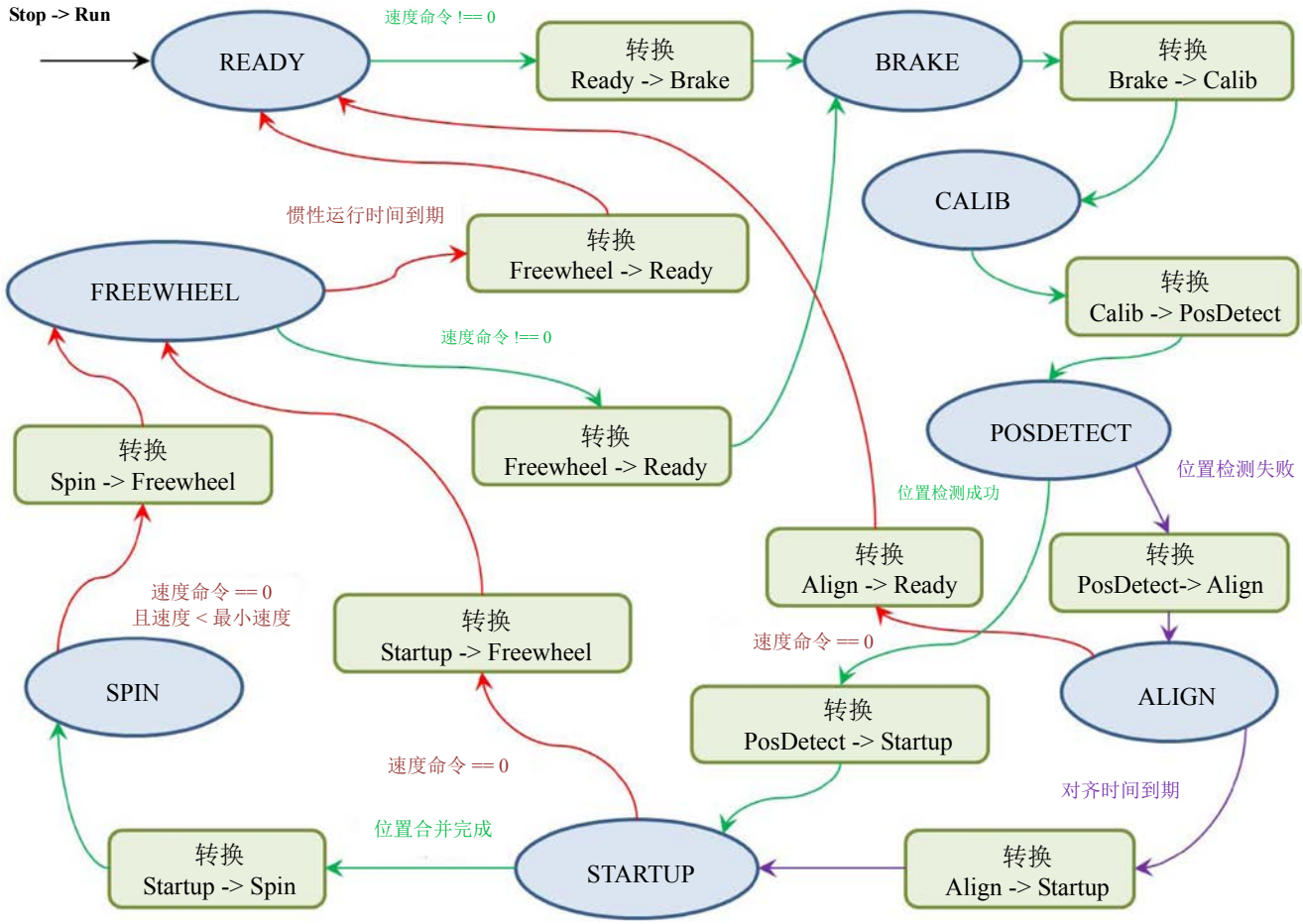

图 4. Run状态下的应用流程

已经声明 Run 子状态机。因此，Run 子状态标识变量定义如下：

```
typedef enum {
      CALIB     = 0,
      READY     = 1,
      BRAKE     = 2,
      POSDETECT = 3,
      ALIGN     = 4,
      STARTUP   = 5,
      SPIN      = 6,
      FREEWHEEL = 7,} M1_RUN_SUBSTATE_T;  /* Run sub-states */
```

正如前面所说，应用不传递 CALIB 子状态。相反，从 STOP 状态转换后，应用直接进入 READY 状态。保留 CALIB 子状态是为了兼容原始无传感器 PMSM FOC 应用。

对于 Run 子状态，定义了如下用户函数集：

```
static void M1_StateRunCalib(void);
static void M1_StateRunReady(void);
static void M1_StateRunBrake(void);
static void M1_StateRunPosDetect(void);
static void M1_StateRunAlign(void);
static void M1_StateRunStartup(void);
static void M1_StateRunSpin(void);
static void M1_StateRunFreewheel(void);
```

Run 子状态的用户瞬态函数原型：

```
static void M1_TransRunReadyBrake(void);
static void M1_TransRunReadyCalib(void);
static void M1_TransRunCalibAlign(void);
static void M1_TransRunBrakeCalib(void);
static void M1_TransRunAlignStartup(void);
static void M1_TransRunAlignReady(void);
static void M1_TransRunCalibPosDetect(void);
static void M1_TransRunPosDetectStartup(void);
static void M1_TransRunPosDetectAlign(void);
static void M1_TransRunStartupSpin(void);
static void M1_TransRunStartupFreewheel(void);
static void M1_TransRunSpinFreewheel(void);
static void M1_TransRunFreewheelBrake(void);
static void M1_TransRunFreewheelReady(void);
```

Run 子状态的函数表初始化：

```
/* Sub-state machine functions field */
      static const PFCN_VOID_VOID mM1_STATE_RUN_TABLE[8] =
      {M1_StateRunCalib,
       M1_StateRunReady,
       M1_StateRunBrake,
       M1_StateRunPosDetect,
       M1_StateRunAlign,
       M1_StateRunStartup,
       M1_StateRunSpin,
       M1_StateRunFreewheel};
```

3.2 制动和转子初始位置检测的软件实现

3.2.1 电机制动

电机的制动算法是在应用状态机中执行，即 RUN 状态的 BRAKE 子状态中执行。在快速控制环内测量 A 相电流。为得到相电流的幅值，需跟踪 A 相电流的最大测量值，并在快速环的每千次转换时评估一次。这样就能保证即便在低速时，也能成功检测到每个周期的相电流幅值。
制动算法所用的变量数据被集成到如下定义的结构之中：

```
typedef struct
      {
        UWord16  uw16BrakeLoopDuration;  /* duration of max current detection window */
        UWord16  uw16BrakeLoopCounter;   /* counter measuring the detection window */
        Frac16   f16MaxIA;               /* detected value of max. phase A current */
      } MCS_PMSM_BRAKE_T;
```

结构成员说明如下：

- 最大电流检测窗口的持续时间（以快速控制环节拍数为单位）
- 存储实际节拍数的计数器
- 实际检测窗口中捕捉到的 A 相最大电流值

3.2.2 转子初始位置检测

为使代码更易于阅读，转子初始位置检测算法分为两个函数：
完成所有扇区中的电流测量后，从状态机调用的函数原型为：

```
void MCS_ MCS_PMSMPositionDetectionInit(MCS_PMSM_FOC_T *psFocPMSM,
MCS_PMSM_POS_DETECT_T *psPosDetect);
```

根据实测电流评估初始位置所用的函数原型为：

```
void MCS_PMSMGetInitialPosition(MCS_PMSM_POS_DETECT_A1_T *psPosDetect)
```

DRM148 中说明的第一个使用 FOC 结构的函数。输入/输出结构指针引用的附加结构定义如下：

```
typedef struct
  {
  GFLIB_RAMP_T sVoltageVectorRampParams; /* Ud Voltage ramp parameters */
  Frac16       f16UdMax;                 /* Maximum applied Ud voltage */
  Frac16       f16UdMin;                 /* Minimum applied Ud voltage */
```

```
Frac16    f16CurrentInSector[6];        /* Array of measured currents in sectors */
Frac16    f16InitPos;                   /* Detected initial rotor position */
Frac16    f16MinCurrentDelta;           /* Minimum difference between currents */
UWord16   uw16PulseCnt;                 /* Counter used for measurement of the turn-on
                                           and turn-off of the PWM outputs*/
UWord16   uw16PulseOnDuration;          /* Turn-on time of the PWM outputs */
UWord16   uw16PulseOffDuration;         /* Turn-off time of the PWM outputs */
Word8     w8MainVector;                 /* Detected main vector of the rotor position */
Word8     w8AuxVector;                  /* Auxiliary vector used for fine tuning */
bool      bPulseOffFlag;                /* Flag used for enabling/disabling of the PMW
                                           outputs */
bool      bCurrentMeasurementCompleted; /* The current measurement completed */
bool      bErrFlag;                     /* The position detection failed */
}   MCS_PMSM_POS_DETECT_T;
```

结构成员说明如下：

- Ud 电压斜率结构—用于控制所施加的 Ud 电压
- 最大 Ud 电压（根据电机最大相电流来选择）
- 最小 Ud 电压—Ud 电压矢量的最小值
- 各 SVM 扇区中测量得到的最大电流数组
- 检测到的初始位置
- 当算法返回初始位置的可靠值时，测得电流之间的最小差值
- 用于测量 PWM 输出的导通和关断时间的计数器变量（以快速控制环节拍数为单位）
- 施加 Ud 电压矢量时，PWM 输出的导通时间
- PWM 输出的关断时间
- 检测到的转子位置主矢量
- 用于精调转子初始位置的辅助变量
- 完成电流测量时的置位标志，以便可以执行位置检测算法
- 错误标志，当测得电流之间的差值太小，导致无法可靠地检测位置时，进行置位

3.2.3 修改位置检测算法以支持任意电机

可以调整转子初始位置检测算法以支持任意电机。唯一的限制条件是电机定子线圈的电气时间常数 $\tau = L_S / R_S$ 必须大于或等于快速（电流）控制环的周期。施加于定子绕组的电压必须足够高，使得流过定子线圈的电流能够引发饱和现象。另一方面，在转子初始位置检测过程中施加的电流不能太高，否则会产生扭矩使电机转动。

要修改支持任意电机的位置检测算法，必须更改以下宏定义：

```
#define CONTROL_FREQ        10000    /* in Hz, frequency of the fast control loop */
#define MAX_U_LENGTH        2        /* in Volts - should be In * R * 0.5 (such a voltage
                                     that the current will be 50 % of nominal value )*/
#define MIN_U_LENGTH        0.1      /* in Volts - should be set to 10-30% of MAX_U_LENGTH */
#define MIN_TO_MAX_TIME     1        /* in ms - should be equal to Tau */
#define MIN_DELTA_CUR       20       /* in mA - should be up to 5% of nominal current */
```

4 MCU 外设

表 1总结了 Kinetis KV10Z32 MCU 上的外设及其在 PMSM 无传感器矢量控制应用中的使用情况。

表1. Kinetis KV10Z32外设概览

Kinetis KV10 外设			应用中使用	用途
组别	模块	模块或通道数		
模拟	ADC0	11 个单端通道，其中可构成 2 个差分对	3 通道	直流母线电压和电机相电流检测
	ADC1	11 个单端通道，其中可构成 2 个差分对	2 通道	
	比较器	2 个模块，各 7 通道	—	—
	DAC	1 个模块	—	—
通信	SPI	1 个模块，4 个片选信号	1 个模块	MOSFET 驱动器配置
	UART	2 个模块	1 个模块	FreeMASTER 通信
	I2C	1	—	—
定时器	FlexTimer	6 通道	6 通道	产生 6 通道 PWM 用于电机控制
		2 通道	—	—
		2 通道	—	—
	PDB	2 通道用于 ADC 触发	2 通道	用于触发直流母线电压和相电流采样
		2 通道用于 DAC 触发	—	—
	LPT	1 个模块	—	—
其他	eDMA	4 通道	—	—

4.1 用于产生 6 通道 PWM 的 FlexTimer0 配置

FlexTimer 模块 (FTM) 是一个包含 2 至 8 通道的定时器，支持输入捕获、输出比较以及产生 PWM 信号功能以实现控制电机和电源管理应用。FTM 时间基准是一个 16 位计数器，可用作无符号或带符号计数器。Kinetis KV10 上有三个 FTM 模块实例。一个 FTM 有 6 个通道，另外两个 FTM 有 2 个通道。

关于如何配置 FlexTimer 以产生中心对齐的 PWM 并插入死区，请参见应用笔记“在 ACIM/PMSM 电机控制应用中使用 FlexTimer”（文档 AN3729）。

由于 AN3729 支持在 ColdFire V1 上实现的早期版本 (1.0) 的 FlexTimer，并且考虑到所用的硬件 (TWR-MC-LV3PH)，具体配置有几个不同之处，说明如下：

- 需要在时钟门控寄存器中使能 FlexTimer 模块的系统时钟：
  SIM→SCGC6 |= SIM_SCGC6_FTM0_MASK;
- 需要禁用某些寄存器的写保护以便进行更新：
  FTM0→MODE |= FTM_MODE_WPDIS_MASK;
- 建议使能 FlexTimer 内部计数器以便在调试模式下运行：
  FTM0→CONF |= FTM_CONF_BDMMODE(3);
- 当硬件调试接口（JLink、Multilink 等）连接到微控制器时，MCU 便处于调试模式。它不取决于运行代码中是否包含断点。
- FlexTimer0 产生的 PWM 信号直接连到 MOSFET 驱动器。

  由于安全原因，塔式低压功率板上使用的 MOSFET 驱动器的上桥臂晶体管的输入信号是低电平有效的。因此，还需要设置 PWM 信号的正确极性：

  FTM0→POL = FTM_POL_POL0_MASK |

  FTM_POL_POL2_MASK |

  FTM_POL_POL4_MASK;
- 占空比通过更改 FlexTimer 值寄存器的值来改变。这些寄存器是双缓存式，要更新其值，不仅需要写入数值，还需要通过加载使能 (LDOK) 位的置位来确认变更。这样可确保所有值同时更新：

  FTM0→PWMLOAD = FTM_PWMLOAD_LDOK_MASK;

  每次更改值寄存器后，都需要写入 LDOK 位，因此不仅要在加载初始值阶段写入，而且在矢量控制算法计算占空比值后的每次更新时也要写入。
- 本应用利用硬件触发模数转换器 (ADC)。来自 FlexTimer 的初始化触发信号 (Initialization Trigger signal) 用作主触发信号，用于触发可编程延迟模块 (Programmable Delay Block)，以便确定启动模数转换的时刻。
  FTM0→EXTTRIG |= FTM_EXTTRIG_INITTRIGEN_MASK;

- 最后，为了将信号引出芯片，必须配置 MCU 的输出引脚。输出引脚的信号分配在引脚控制寄存器中进行设置。可用信号列在“无位置传感器的 BLDC 电机的新型启动方法”的“信号复用”一章中，并且与封装相关。（参见第12章节“参考文献”）

```
PORTC->PCR[1] = PORT_PCR_MUX(4);          // FTM0 CH0
PORTE->PCR[25] = PORT_PCR_MUX(3);         // FTM0 CH1
PORTC->PCR[3] = PORT_PCR_MUX(4);          // FTM0 CH2
PORTC->PCR[4] = PORT_PCR_MUX(4);          // FTM0 CH3
PORTD->PCR[4] = PORT_PCR_MUX(4);          // FTM0 CH4
PORTD->PCR[5] = PORT_PCR_MUX(4);          // FTM0 CH5
```

应用代码中实现的端口设置会和基于塔式系统模块构建的硬件解决方案相对应。

4.2 ADC 和 PDB 模块配置

片上 ADC 模块用于对执行矢量控制算法所必需的反馈信号（电机相电流和直流母线电压）进行采样。可编程延迟模块与 ADC 密切合作，触发硬件进行采样。

为了获得额定精度，ADC 模块在应用之前需要执行自校准程序。校准过程还需要编程器干预，以便产生同相端和反相端增益校准结果，并在校准功能完成后将其存储在 ADC 同相端和反相端增益寄存器中。两个 ADC 模块都必须执行校准。校准之后，ADC 模块配置为 12 位精度。CPU 频率设置为 75 MHz，然后利用可用的预分频值，将 ADC 模块的输入时钟设置为 12.5 MHz。该设置对应的转换时间为 2.64 μs（33 个 ADC 时钟周期）。最后，必须在 ADC 的状态和控制寄存器 2 (Status and Control Register 2) 中使能硬件触发。

可编程延迟模块 (PDB) 在从内部/外部触发信号或可编程间隔节拍信号到 ADC 的硬件触发输入信号之间提供可控延迟，从而以精确的时序执行各次 ADC 转换。PDB 模块内置一个计数器，计数到模值 (Modulus value) 时溢出。由于 PDB 的输入触发信号由 FTM0 周期性提供，因此 FTM0 的模寄存器的值可以设置为 PDB 的模寄存器值。通道延迟寄存器 (Channel Delay registers) 中的值用于确定 ADC 的触发时刻，以便启动直流母线电压采样和电机相位的模数转换。KV10 MCU 上的 PDB 模块支持 15 种不同的输入触发源，参见“KV10 子系列参考手册”（文档 KV10P48M75RM）“PDB 配置”部分的“芯片配置”一章中的列表。同样，和 FTM0 类似，为了响应延迟寄存器中所做的更改，必须置位 LDOK 位。

KV10 MCU 上的 PDB 时钟源与 FTM 时钟源不同：PDB 和 ADC 使用总线时钟，FTM 使用系统时钟。

4.3 ADC 转换时序、电流和电压采样

当 FlexTimer0 的计数器在溢出到初始值之后复位时，FlexTimer0 触发内部硬件信号。该信号进入可编程延迟模块 (PDB)，经过预定延迟后，触发电压和电流的模数转换。Kinetis KV10 75 MHz MCU 上有两个 ADC 模块。各 ADC 模块与 PDB 模块的一个通道关联。每个 ADC 模块有两个结果寄存器（2 通道），对应于 PDB 通道的两个可编程预触发延迟。因此，可以执行四次模数转换而无需请求中断（DMA 不用于数据传输时）。本应用中，在 CPU 不干预的情况下，

只需触发 3 次转换（两个电机相电流和一个直流母线电压）。图 5 时序图显示了模块之间的相互连接和 ADC 中断产生情况。

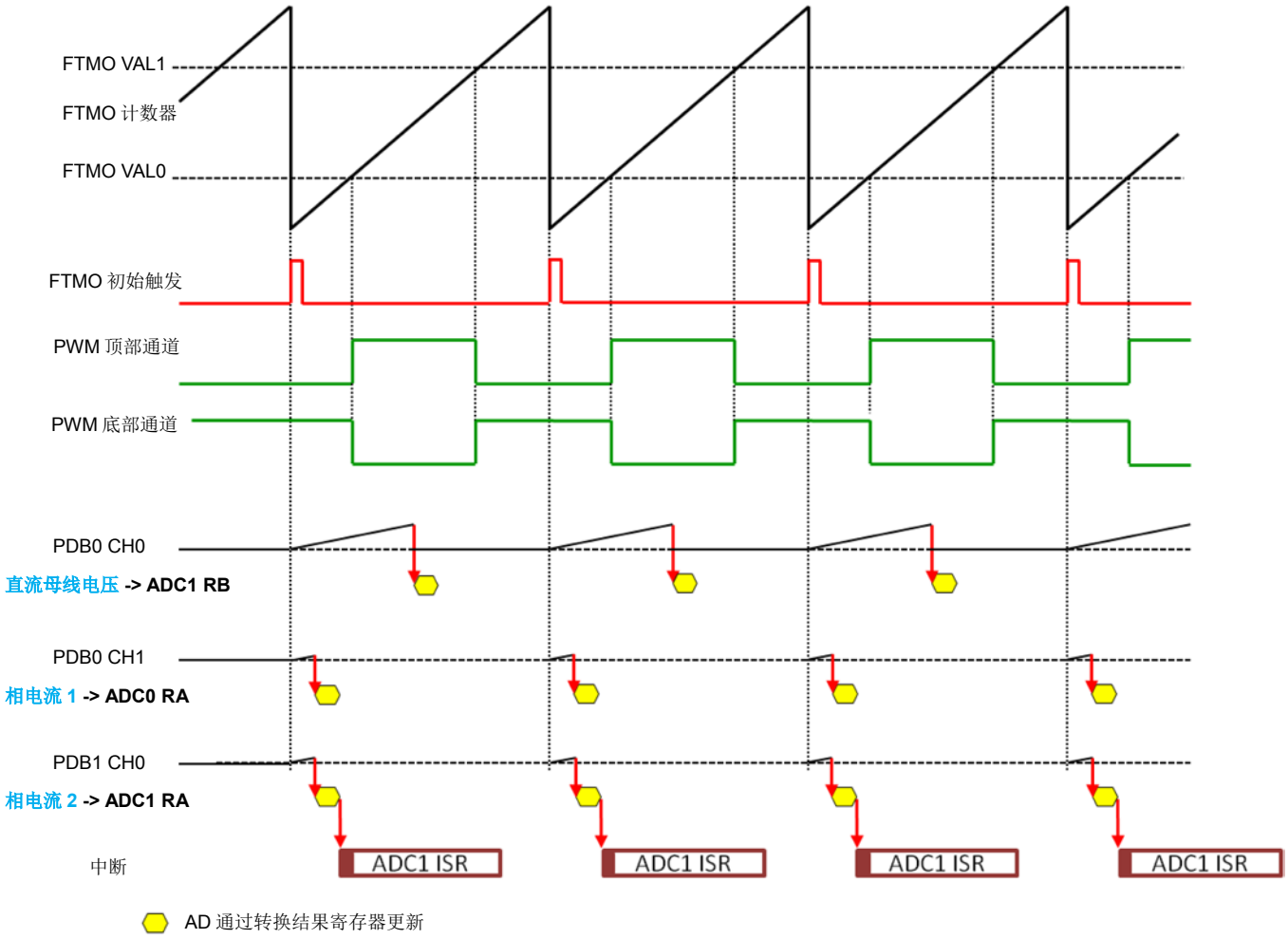

图 5. ADC转换时序图

4.4 电流测量

与 ADC 转换触发时序密切相关的是分配到 ADC 通道进行测量的模拟信号。对于 FOC 快速（电流）控制环的计算，必须知道电机所有三个相电流的值。由于只有两个 ADC 模块，因此在某一时间可以仅对两个模拟量进行采样。假设电机是一个对称三相系统，那么所有三个瞬时相电流之和为 0。

$$0 = i_A + i_B + i_C \qquad \textit{公式 1}$$

在下桥臂晶体管导通的时刻测量相电流，如果占空比很高（电流值在正弦曲线的最大值区域），那么可以测量电流的时间将非常短。为了获得稳定的分流电阻压降，下桥臂晶体管必须导通至少一个临界脉冲宽度。通道根据产生定子电流空间矢量的扇区来选择。该分配在 ADC1 中断服务例程结束时执行。因此，采样两个相电流即足够，第三个很容易根据公式 2来计算：

$$\begin{aligned} &\text{Sector 1,6: } i_A = -i_B - i_C \\ &\text{Sector 2,3: } i_B = -i_A - i_C \\ &\text{Sector 4,5: } i_C = -i_B - i_A \end{aligned} \qquad \textit{公式 2}$$

图 6 显示了两种情况（第一种情况是 30°，第二种情况是 60°），用以说明为什么需要计算第三个电流。

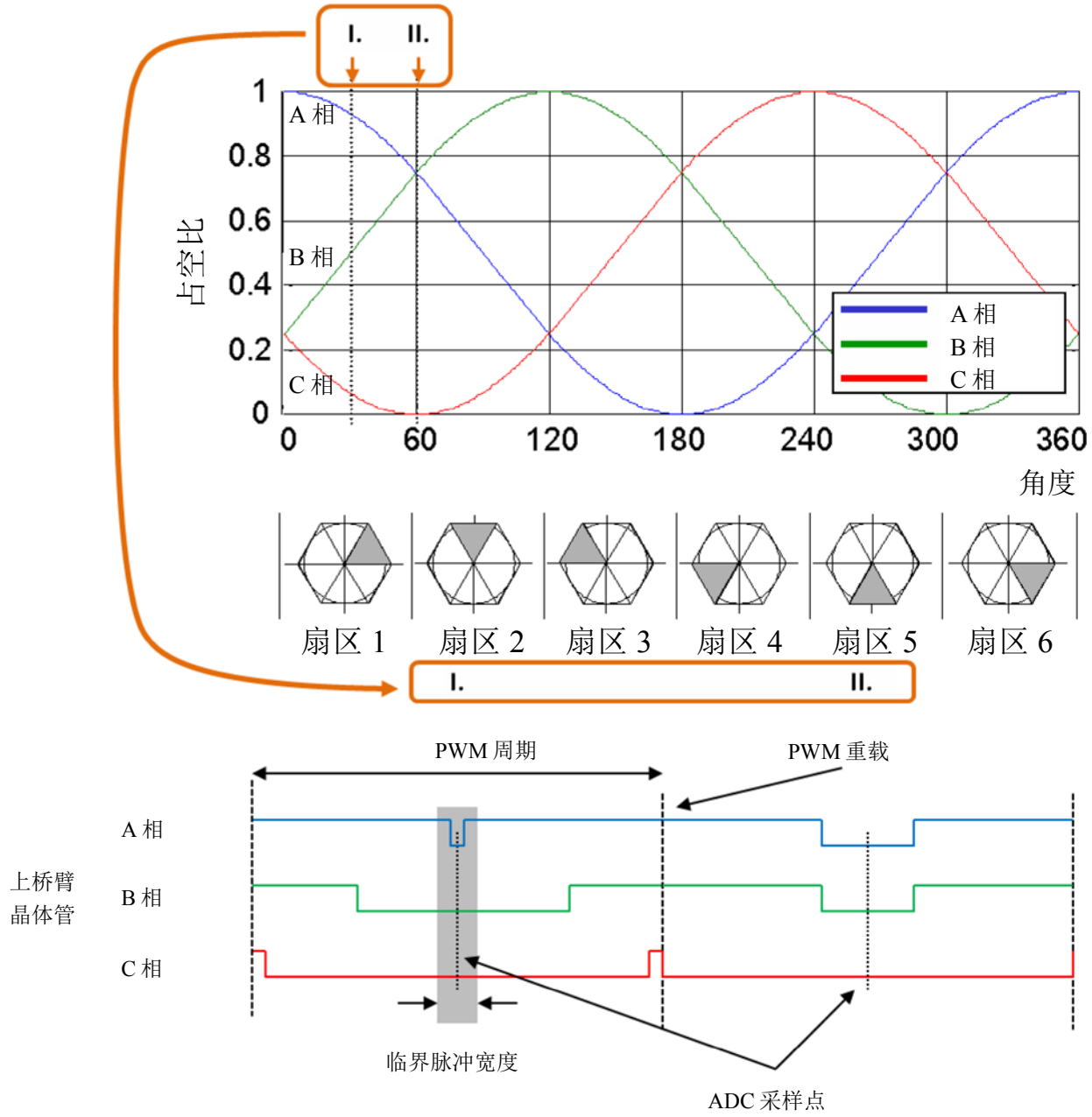

图 6. 电流采样

如上图所示，在 60°时，可以对所有三个电流进行采样，因为前面说过，电流是在下桥臂晶体管导通时进行采样。因此，脉冲宽度足以使电流稳定下来，并由模数转换器采集信号值。在 30°时，脉冲太短，因而无法对 A 相电流进行采样。

4.5 SPI 配置

应用中使用 SPI 接口来实现 MOSFET 栅极驱动器 MC33937 与 KV10 MCU 之间的通信。MC33937 栅极驱动器置于塔式低压功率板上，用于驱动三相逆变器的高端和低端 MOSFET 晶体管。在应用中，必须执行 MC33937 初始化，主要是设置死区时间。在电机运行期间，也会定期检查驱动器的状态寄存器，以便获得关于锁存故障的信息。MC33937 驱动器要求 SPI 信号

具有精确的时序。不能使用 MCU 的 SPI 模块的默认设置。SPI 信号的确切时序参见“三相场效应晶体管前置驱动器数据手册”（文档 MC33937）。

4.6 SCI (UART) 配置

使用 SCI 来实现主机系统与嵌入式应用之间的通信。主机系统是指笔记本电脑或 PC，其中安装了 FreeMASTER 软件来控制应用并以可视化方式显示其状态。Kinetis KV10 上有两个 UART 模块。由于硬件解决方案基于塔式模块，因此使用 UART1。受限于 OpenSDA – CDC 串行通信驱动器的使用，通信速度设置为 19200 Bd。如果 PC 与嵌入侧之间使用直接 RS232 通信，允许将通信速度提高到 115200 Bd。

5 中断

由于 MCU 可通过硬件触发模数转换，因此该应用只需非常少的中断。

5.1 ADC1 中断

当 ADC1 模块的通道 A 转换完成且具有最高优先级时，便会触发该中断请求。中断产生期间，物理量的采样值已在以下寄存器中准备就绪：ADC0 的结果寄存器 A（电机相电流 1）、ADC1 的结果寄存器 A（电机相电流 2）和 ADC1 的结果寄存器 B（直流母线电压）。只能允许一个模块使能该中断，否则会同时产生两个中断请求，因为电机相电流的采样触发信号是同时产生的。ADC1 中断服务例程(ISR)开始执行时会调用“应用状态机(Application State Machine)”函数。如果应用处于 Run 状态，则执行 PMSM 矢量控制算法的快速（电流）控制环，包括位置和转速估算。慢速（转速）控制环的频率由快速控制环中的一个软件计数器来决定；每执行一次快速控制环，软件计数器的值便递减。读取触发该中断的 ADC 通道的结果寄存器，可清除中断标志。因此，在每个特定状态机函数开始时，都需要读取模数转换结果，即使这些值只有在以后的程序执行中才会用到。

图 7所示的流程图概述了应用处于 Run 状态和 Spin 子状态时 ADC 中断服务例程执行过程中的程序流程。

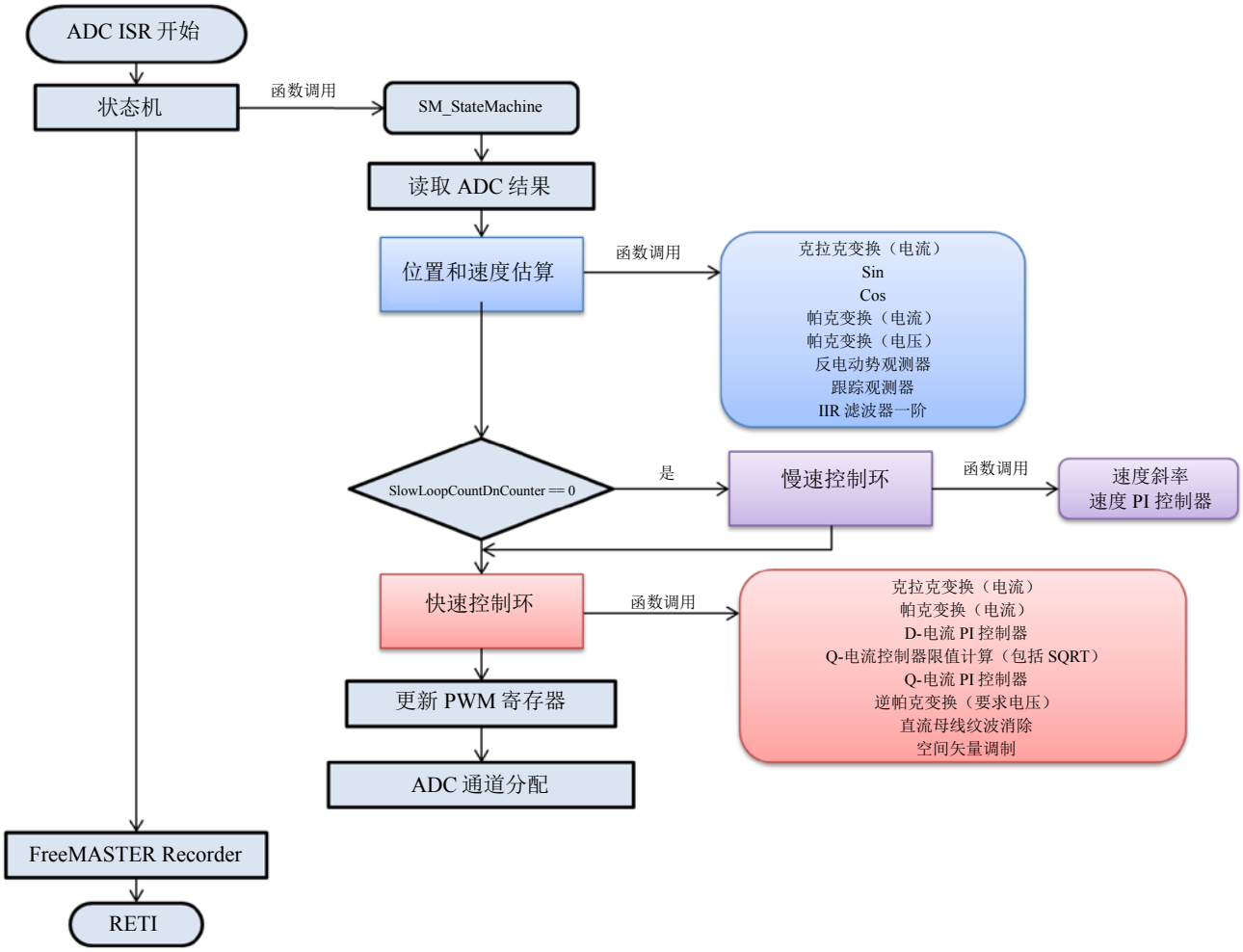

图 7. ADC ISR流程图

5.2 端口中断

只要按下 KV10 塔式上的 SW2 按钮，就会产生端口中断，用户按钮事件的处理在端口中断相关的 ISR 中执行。开始时，中断标志清零。按 SW2 按钮就会使能/禁用演示模式。第一次按 SW2 按钮会将应用状态置于 RUN 模式，所需的速度根据预定义配置文件分若干步改变。再按 SW2 按钮会使应用返回 STOP 模式。

关于应用控制的更多信息参见第9章节“应用程序运行”。利用 FreeMASTER 控制界面可以更好地控制和诊断。

5.3 PDB 错误中断

PDB 错误 ISR 用于清除以下情况下产生的序列错误故障：PDB 启动 ADC 采样，但 ADC 模块的特定 **ADC*x*_SC1*n*** 寄存器中的 COCO 标志未被清除。由于下面所述的原因，结果寄存器中的值未被读取。

- PDB 模块随后重新启动。PDB 产生的触发信号周期与 ADC 转换完成中断的周期 PWM 周期相同。如果用户在代码中设置一个中断，它将停止代码执行。但是，即使程序执行停止，PDB 也会产生下一转换的触发信号。COCO 标志未被清除，PDB 产生序列错误。
- 另一种情况是 ADC 转换完成中断的执行（计算快速控制环时）时间超过一个 PWM 周期。如果用户在 ADC 转换完成中断中加入其他任务，就会发生这种情况。除了产生 PDB 序列错误以外，更严重的影响是在控制过程的质量方面，因为有一个重要假设未得到满足：控制算法的执行需要小于采样周期。实时控制应用的设计必须保证这种情况绝不会发生。

6 项目文件结构

本项目中的源文件(*.c)和头文件(*.h)总数超过一百。因此，本文仅详细说明关键的项目文件，其余文件则分组说明。

主项目文件夹分为三个目录：

- **build\iar\kv10\PMSM_Sensorless—**包含 IAR 编译器的配置文件以及编译器的输出可执行文件和目标文件。如果您的计算机上安装了 IAR Embedded Workbench for ARM，双击位于目录\build\iar\中的工作空间文件 PMSM_SAC_SENSORLESS.eww 就会启动 IAR IDE。
- **freemaster\PMSM_Sensorless—**包含 FreeMASTER 配置文件 PMSM_FOC_KV1x.pmp 和支持文件（HTML 格式的控制页面和带变量地址的二进制文件）。

  它还包含用于“电机控制应用精调工具（Motor Control Application Tuning Tool，MCAT）”PMSM_FOC_KV1x_MCAT.pmp 的 FreeMASTER 项目。
- **src—**包含项目源文件和头文件，其内容说明如下：

  src\projects\kv10\PMSM_Sensorless 文件夹中的文件：

  — **main.c** 和 **main.h** 包含基本应用初始化（使能中断）、访问 MCU 外设的子程序和中断服务例程。FreeMASTER 通信在后台无限循环中执行。

  — **state_machine.c** 和 **state_machine.h** 包含应用状态机结构定义，处理应用状态之间的切换和应用状态跃迁。

  — **motor_structure.c** 和 **motor_structure.h** 包含专门用来电机控制算法（矢量控制算法、位置和转速估算算法、速度控制环）执行的结构定义和子程序。

  — **M1_statemachine.c** 和 **M1_statemachine.h** 包含应用在特定状态或状态迁移时执行的软件例程。

  — **freemaster_cfg.h** 是 FreeMASTER 界面的配置文件。

— **PMSMFOC_appconfig.h** 包含应用控制过程的常量定义（电机和稳压器的参数以及其他矢量控制相关算法的常量）。利用“电机控制应用精调工具 (MCAT)”调整应用以支持其他电机时，此文件由该工具在参数设置过程结束时产生。

— **\peripherals**文件夹包含应用所用外设 (FlexTimer、ADC、PDB、SPI、PIT) 的静态配置相关的重要文件。

src\文件夹中的子目录：

— **\common**和**\cpu**文件夹包含 CPU 初始化例程。

— **\cpu\isr.h** 是一个重要文件，包含分配给中断向量的外设中断服务例程的定义。在该文件中，用户可以为其他外设中断添加 ISR 定义。

— 子目录中的**\drivers**文件包含 UART 和看门狗配置的通用源文件和头文件，以及 CPU 时钟设置程序。

— **\platforms\tower.h** 包含 Kinetis 塔式卡定义（CPU 速度和 UART 参数）。

src\cpu\headers 文件夹中的文件：

— **MKV10Z7.h** 头文件包含 MCU 所有寄存器和寄存器各位的宏定义。

src\MMCLIB\文件夹中的文件：

— **CM0+_MMCLIB_IAR.a** 软件库包含电机控制、通用数学和滤波器算法。该文件夹及子文件夹中的其他文件是相关的头文件，每个头文件对应于库中的一个特定函数。

— **ACLIB\CM0+_ACLIB_IAR_r0.2.a** 包含用于估算转子位置和转速的高级控制算法（反电动势观测器和跟踪观测器）。

src\文件夹中的其他子目录：

— **\FreeMASTER** 包含 FreeMASTER 应用程序的所有源文件，无需访问它或更改其中的任何内容。编程器接口仅通过 src\projects\kv10\PMSM_Sensorless 文件夹中的 **freemaster_cfg.h** 文件提供。

— **\SAC**文件夹（传感器和执行器组件）包含用于访问外设的例程，电机控制算法使用这些外设来检测输入反馈物理量（电流、电压、转速、位置），并根据计算输出变量（FlexTimer、MOSFET 前置驱动器）设置执行器。

7 存储器使用情况

表 2总结了芯片存储器使用情况。

表2. 存储器使用情况（数值单位为字节）

存储器	Kinetis MKV10Z32 总计提供	应用使用
程序 Flash（应用代码）	32 KB	20 460 B
数据 Flash（应用常数）		672 B
数据 RAM（应用变量）	8 KB	2 356 B

8 硬件设置

塔式模块式开发系统用作基于 Kinetis KV10 的 PMSM 无传感器控制的硬件平台。它包含下列模块：

- 塔式侧板模块 (TWR-ELEV)
- Kinetis K60 塔式模块 (TWR-KV10Z32)
- 带电机的三相低压功率模块 (TWR-MC-LV3PH)

塔式系统的所有模块都可以通过 Freescale 网站或代理商订购，因此用户可以轻松构建目标应用所需的硬件平台。

8.1 硬件设置和跳线配置

利用塔式系统的模块构建该系统并不困难。外设模块和 MCU 模块插入侧板连接器，模块板侧的白条决定其与功能侧板（带 mini-USB 连接器、电源和开关的侧板）的连接方向，如图 8所示。

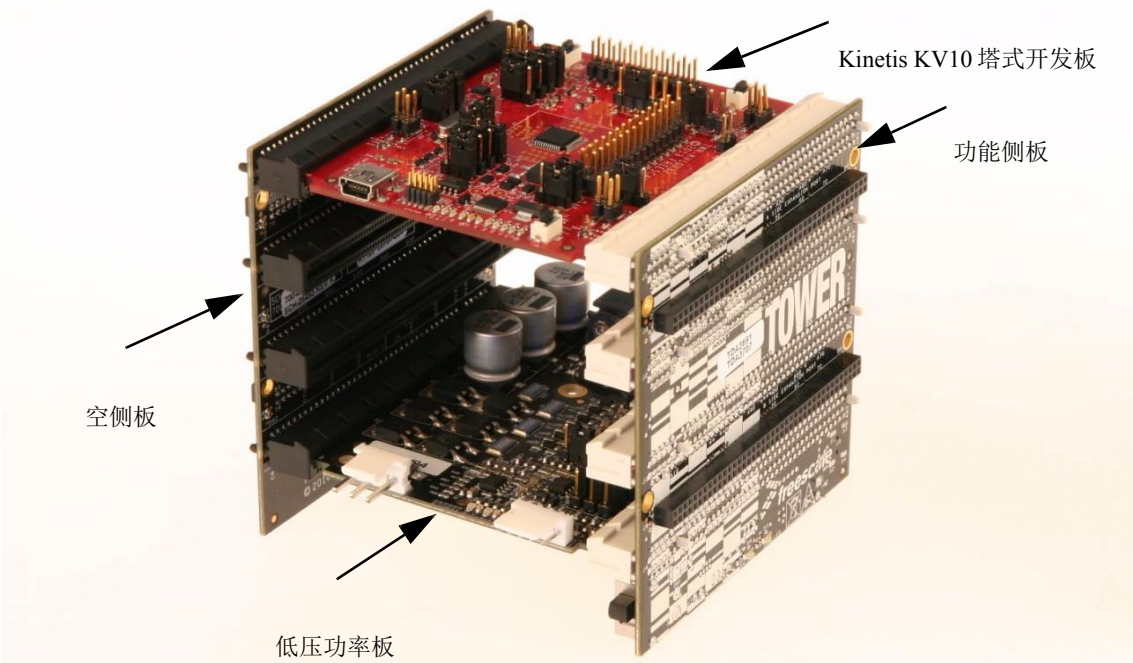

图 8. 基于塔式系统模块的硬件

塔式 KV10 MCU 模块和塔式三相低压功率板上的跳线都需要配置。

塔式 KV10 板的跳线设置如表 3所示。有关跳线的详细说明，请参见“TWR-KV10Z32 用户指南”。

表3. TWR-KV10Z32板的跳线设置

跳线编号	设置	跳线编号	设置
J1	2-3	J14	开路
J2	1-2	J18	2-3
J3	2-3	J19	2-3
J4	1-2	J20	2-3
J5	1-2	J21	3-4
J7	1-2	J22	3-4
J8	2-3	J25	开路
J9	1-2	J26	1-2
J10	2-3	J27	1-2
J11	开路	J28	1-2
J12	开路	J29	1-2
J13	开路	—	—

跳线设置如表 4所示，图 9突出显示了跳线位置。有关塔式低压功率板的更多详情（比如：硬件过流的阈值设置），另请参见“TWR-MC-LV3PH 用户指南”（文档 TWRMCLV3PHUG）。

表4. TWR-MC-LV3PH板的跳线设置

跳线编号		设置	注释
J2	VDDA 源选择	1-2	内部模拟电源
J3	VSSA 源选择	1-2	内部模拟电源
J10	AN6 信号选择	1-2	C 相电流信号
J11	AN5 信号选择	1-2	B 相电流信号
J12	AN2 信号选择	1-2	A 相电流信号

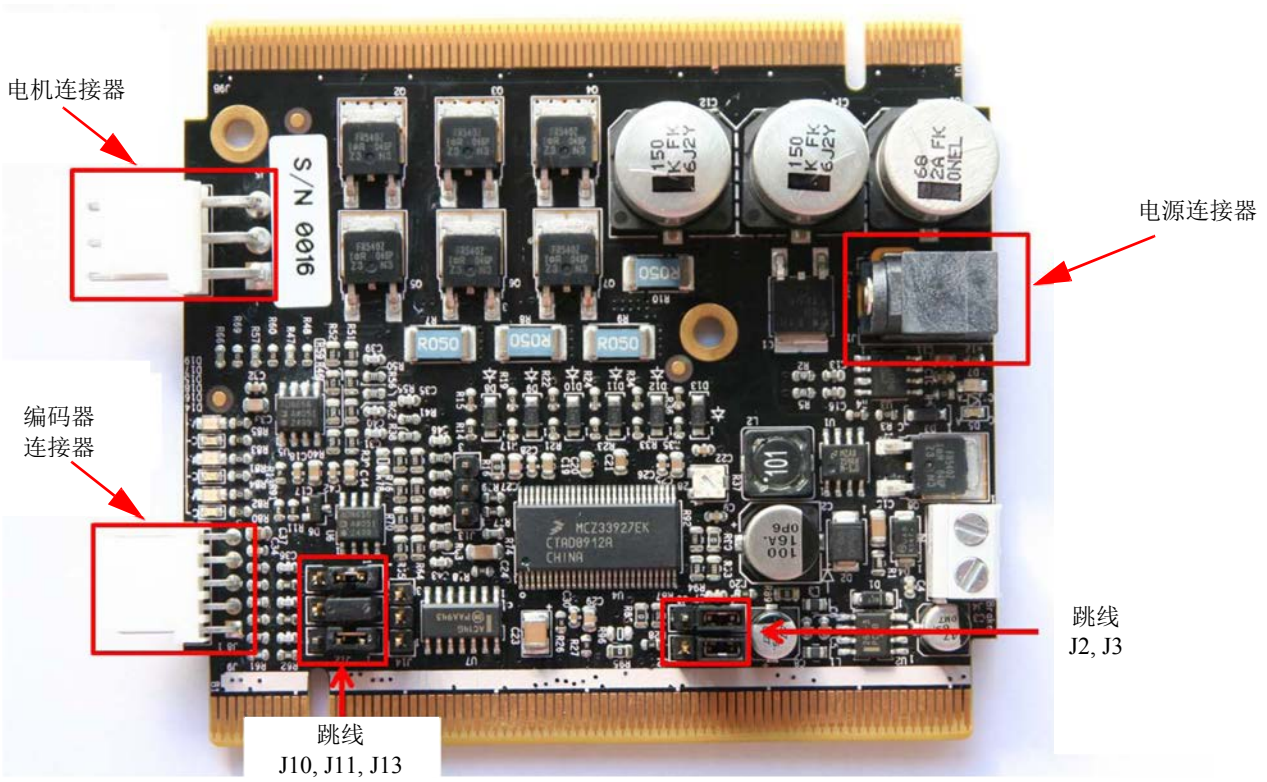

图 9. TWR-MC-LV3PH上的跳线和连接器位置

表 5显示了 TWR-MC-LV3PH 的电机连接器的信号分配。

表5. TWR-MC-LV3PH上的电机连接器

连接器	引脚编号	说明
电机连接器 J5	1	电机 A 相
	2	电机 B 相
	3	电机 C 相

参考设计使用的电机是 TWR-MC-LV3PH 套件的一部分。它是一款 BLDC 电机，反电动势电压呈梯形，定子上有凸极。这种形状与 PM 同步电机的不同之处是后者定子上的绕组是分布式，形成正弦波形磁场。两类电机的转子结构相同（轴上有凸极）。矢量控制算法最初是针对 PM 同步电机而开发的，它假设磁场为正弦形状，BLDC 电机也可以采用相同的控制策略。性能不是最佳，但驱动产生的音频噪声小于传统的六步换向控制。其主要优势是客户可以在高性价比硬件解决方案上学习并采用无传感器矢量控制。

表 6提供了电机规格。

表6. 电机规格

电机规格	
电机整体规格	
制造商名称	Linix
型号	45ZWN24-40
标称电压（线间）	24V DC
标称转速	4000 rpm
额定功率	40 W
特定型号规格	
定子绕组电阻（线间）	1 Ohm
定子绕组电感 d 轴	426 µH
定子绕组电感 q 轴	460 µH
极对数	2
反电动势常数 k_e	0,01456 $V.s.rad^{-1}$

注意

应用参数（速度 PI 控制器和启动电流值）针对轴上安装有塑料圈（套件的一部分）的电机而设置，其他情况下可能发生速度振荡。

9 应用程序运行

利用 TWR-KV10 板上的用户按钮（如第 5.2 节“端口中断”所述）或通过 FreeMASTER 软件，可以运行本应用程序；FreeMASTER 软件还支持变量可视化。FreeMASTER 应用程序由两部分组成：用于变量可视化的 PC 应用程序和在嵌入式应用中运行的软件驱动包。PC 与嵌入式应用程序之间的数据通过 RS232 接口传送。

9.1 在 PC 或笔记本电脑上安装 FreeMASTER

FreeMASTER PC 应用程序可从 Freescale 网页www.freescale.com/freemaster下载。进入页面后，点击“下载”选项卡，然后选择 *FreeMASTER 1.4 Application Installation*。下载 FreeMASTER 应用程序需要注册，必须创建一个帐户后才能登录。登入系统之后，会出现许可协议。您应当阅读许可协议，然后必须点击“我接受”按钮接受该协议。如果使用的是 Internet Explorer，网页顶部会出现信息提示栏，要求您授权文件下载。点击信息提示并选择“下载文件”。随即出现一个对话框，您可以选择“运行”或“保存”。无论何种情况，安装档案都会存储到您的计算机上。若选择“保存”，您可以选择安装档案的保存位置，否则它将被保存到由系统创建的临时文件夹下。库安装档案随后就会下载到您的计算机中。

要执行安装，请点击“运行”按钮。按照屏幕上的提示完成安装过程。

9.2 建立 PC 与嵌入式应用程序之间的连接

FreeMASTER 支持在嵌入式应用程序与 PC 或笔记本电脑之间使用多种通信接口（UART (RS232)、CAN、以太网、USB、BDM 等）。本应用程序采用 RS232，因为串口数据传输的软件开销最低。如今的笔记本电脑不配备 COM 端口，为此，Kinetis KV10 塔式模块配置了 USB 转 RS232 接口（OpenSDA – CDC 串行端口）。通过 USB 线将 Kinetis KV10 塔式模块与笔记本电脑相连，便可在 Windows 系统中建立一个虚拟串行端口。

9.3 FreeMASTER 应用程序运行

要运行 FreeMASTER 应用程序，请双击*\freemaster* 文件夹中的 **PMSM_FOC_KV1x.pmp** 文件。FreeMASTER 应用程序随即启动，并且会自动创建*.pmp 文件所定义的环境。

9.3.1 设置通信

当笔记本电脑通过 USB 线缆连接到 Kinetis KV10 塔式模块时，操作系统就会将 COM 端口号分配给 OpenSDA – CDC 串行端口。此端口号随机分配，因此每次建立连接时，都要设置正确的通信端口（重新插上 USB 线缆可能会分配不同的端口号）。

要设置端口号，请点击菜单项 ***Project* \ *Options …***

然后在打开窗口的 **Comm** 选项卡中指定端口号。

如果端口号选择正确，可用串行端口号列表框旁边会显示文本 *OpenSDA – CDC Serial Port*。

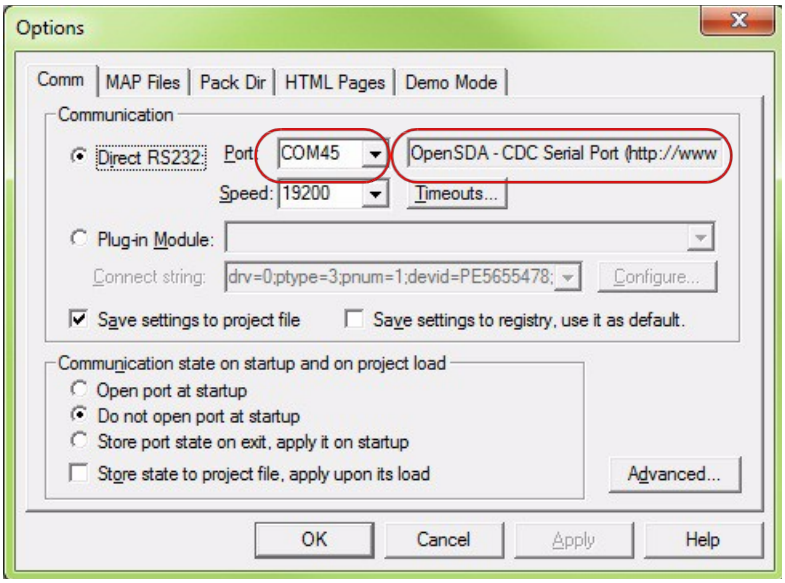

图 10. FreeMASTER通信设置

9.3.2 应用程序运行

9.3.2.1 启动/停止通信

通信设置就绪后，便可启动 PC 与嵌入式应用程序之间的通信。点击 FreeMASTER 工具栏中的 STOP（停止）按钮，如图 11所示。

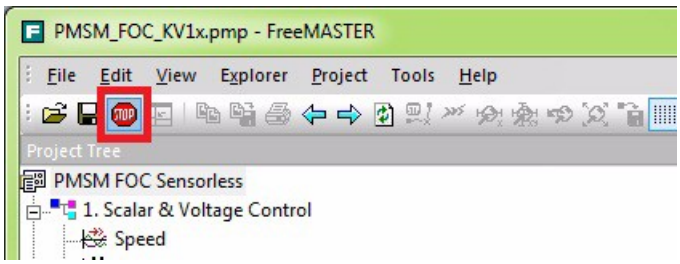

图 11. 启动与嵌入侧的通信

9.3.2.2 启动/停止应用程序、所需速度设置

下一步是将应用程序切换到 RUN 状态。在 FreeMASTER 窗口的 Variable Watch（变量观测）网格中，点击 *Application Switch*（应用程序切换）变量名旁边的下拉列表，选择 ON（开启），如图 12所示：

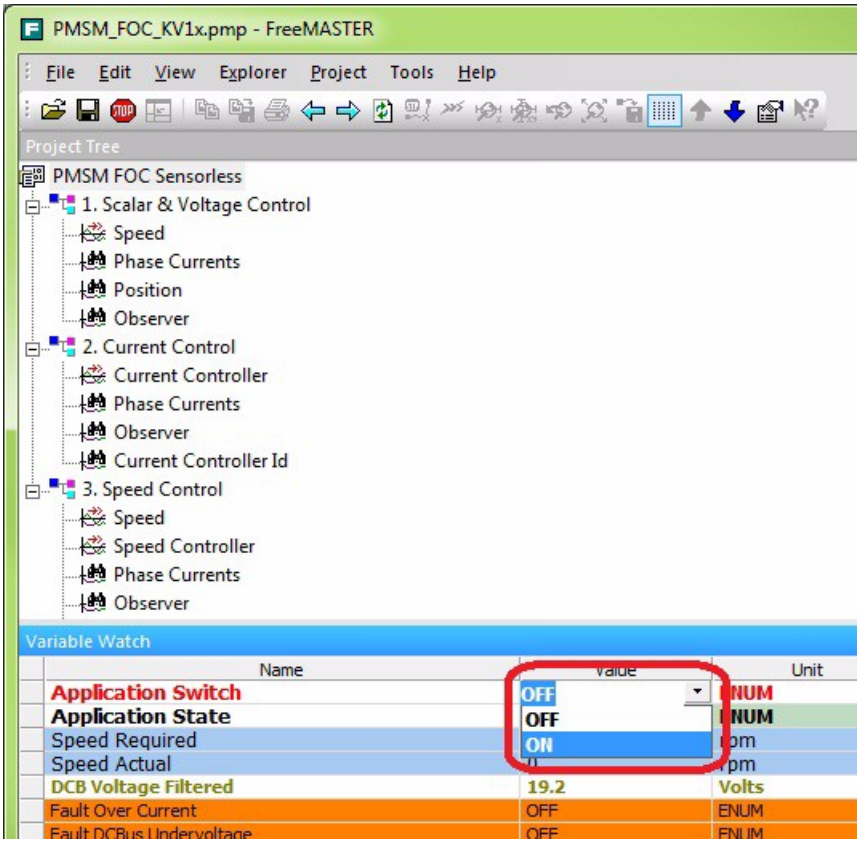

图 12. 启动应用程序

应用程序设置为 Run 状态之后，便可将所需的速度更改为某一非零值。过程与上述步骤相似，即在 Variable Watch 网格的 *Speed Required*（所需速度）变量名旁边输入一个正值或负值。

9.3.2.3 从控制页面运行应用程序

也可以从控制页面运行应用程序，如图 13所示。

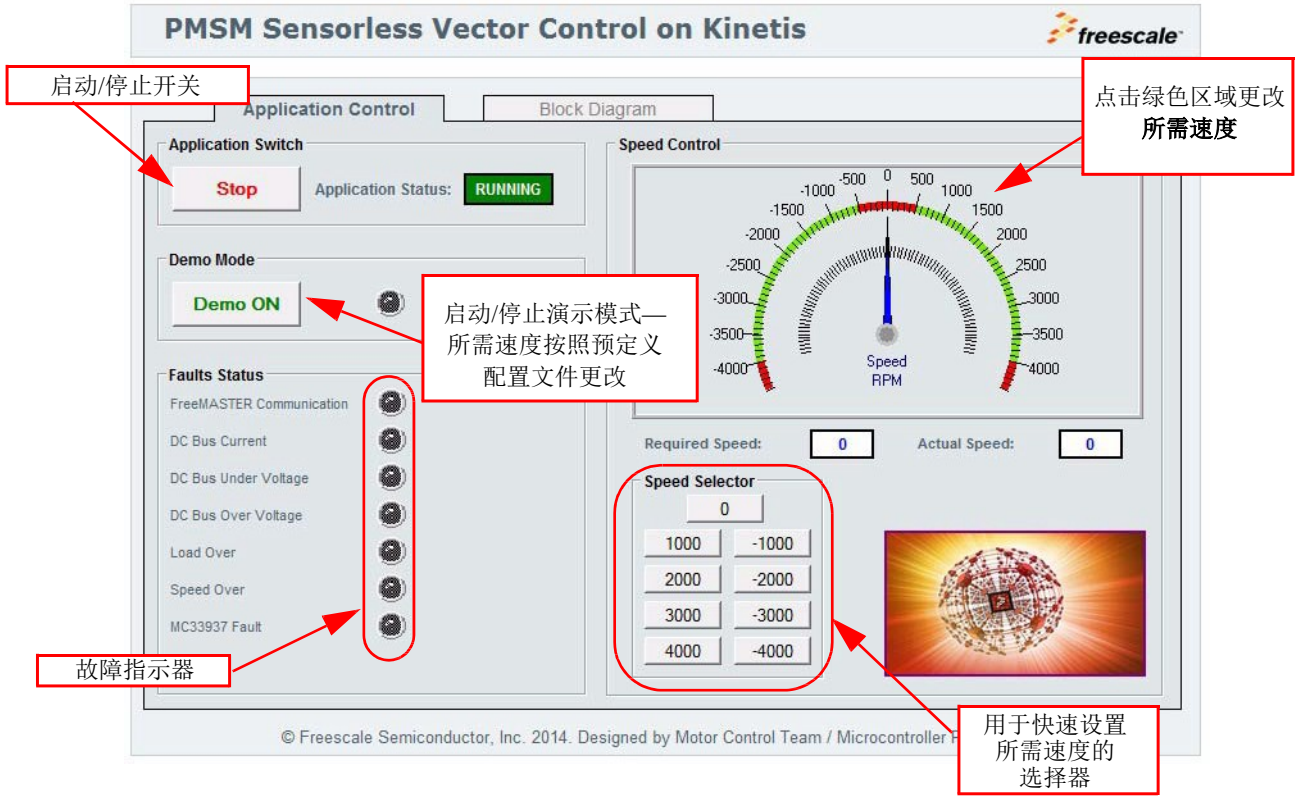

图 13. PMSM无传感器矢量控制页面

9.3.2.4 控制模式选择器

本应用程序可调整支持任何电机的 PMSM 无传感器应用。为此，矢量控制应用分为四个部分（模式），支持通过若干步骤来调整应用，每一步仅调整少量未知参数。为在这些模式之间切换，使用一个名为 *MCAT Control*（MCAT 控制）的变量。有关调整支持任何电机的应用的更多信息，请参见“使用 MCAT 工具调整三相 PMSM 无传感器控制应用”（文档 AN4912）。

位置检测算法仅在 *Speed FOC*（速度 FOC）控制模式下使能。在其他控制模式下，初始位置由对齐确定。电流通道失调校准在对齐子状态之前执行。

9.3.2.5 Scope 和 Recorder

FreeMASTER 应用程序的主要优势之一是变量值的实时可视化。可视化有两种方式，用户可以选择 Scope 或 Recorder。Scope 是指数据流实时连续下载，Recorder 则是将数据存储在嵌入式 MCU RAM 的缓冲区，满足触发条件后，数据以块形式通过通信接口传输，并在 FreeMASTER 窗口中可视化。Scope 的采样周期受限于通信接口的速度，因此用于缓慢改变转速等量。类似示波器的 Scope 的采样周期为微秒级，支持相电流或占空比等快速变化量的可视化。本应用中，Recorder 缓冲区在每次执行快速控制环时更新，即每 100 µs 更新一次。

图 14 显示的是应用程序在 *Speed FOC* 控制模式下运行时可用的 Recorder 和 Scope。

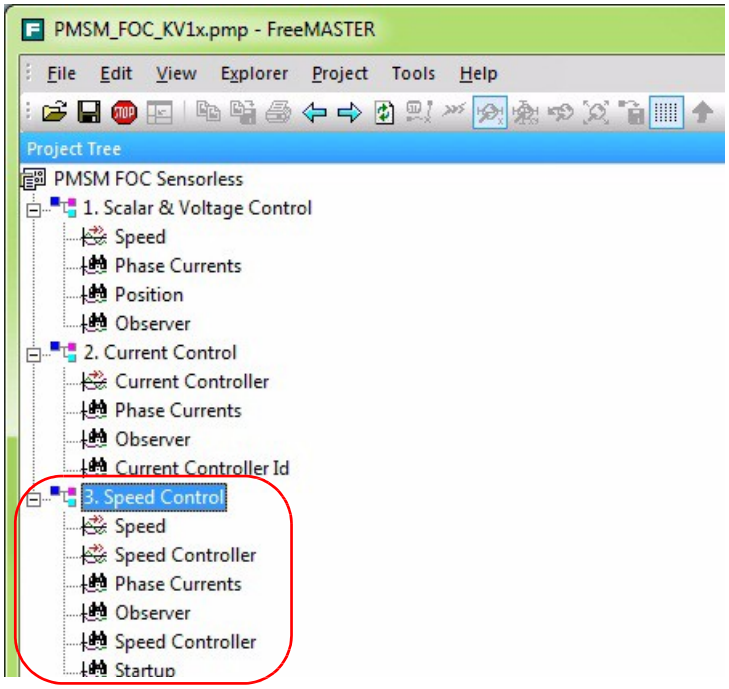

图 14. Speed FOC控制模式下使用的Scope和Recorder

10 测量结果

10.1 CPU 负荷和执行时间

CPU 负荷主要受 ADC1 ISR 的执行影响，其中要执行应用状态机以及 PMSM 矢量控制的快速（电流）和慢速（转速）控制环的计算。

完整的 ADC1 ISR 需要 5209（状态机和快速控制环）到 6036（采用慢速控制环计算）个机器周期。当更新占空比值时，ADC1 中断周期与 PWM 重载事件的频率周期相同。

本应用中，ADC ISR 每 100 µs 产生一次，对应于 10 kHz 的 PWM 频率。对于 75 MHz 的 Kinetis KV10 器件，它消耗 69–80%的 CPU 性能。

10.2 FreeMASTER 结果

图 15提供的屏幕截图用于显示初始位置检测方法的工作原理。不同的截图具有不同的初始位置值。FreeMASTER 截图反映的是各扇区中测量的相电流最大值。

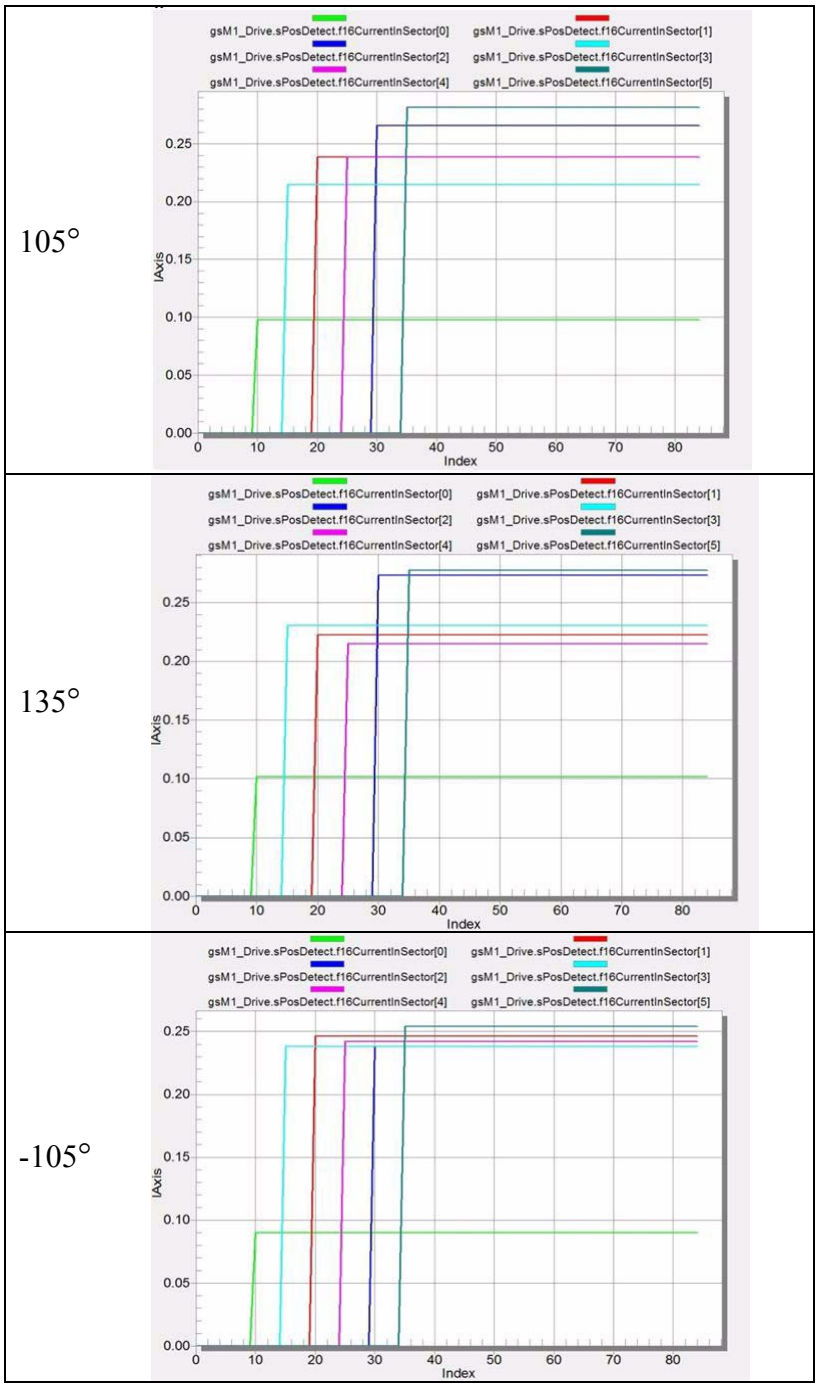

图 15. 通过Recorder在初始位置检测期间获得的相电流最大值

11 结论

执行时间测量结果显示，Kinetis KV10 微控制器只能实现低动态响应的 PMSM 无传感器矢量控制算法，因为没有足够的 CPU 处理能力来执行其他任务。本文提出的解决方案提供抗风算法和转子初始位置检测功能，非常适合吊式风扇应用。

12 参考文献

A New Starting Method of BLDC Motors Without Position Sensor, by Wook-Jin Lee, Seung-Ki Sul, IEEE Trans. on Ind. Electronic, Vol. 42, no. 6, November / December 2006, pages 1532–1538

Sensorless PMSM Field-Oriented Control, Freescale Semiconductor（文档 DRM148）

Kinetis V Series KV10, 32/16 KB Flash Data Sheet, Freescale Semiconductor（文档 KV10P48M75）

KV10 Sub-Family Reference Manual, Freescale Semiconductor（文档 KV10P48M75RM）

Three Phase Field Effect Transistor Pre-driver Data Sheet, Freescale Semiconductor（文档 MC33937）

TWR-MC-LV3PH User's Guide, Freescale Semiconductor（文档 TWRMCLV3PHUG）

Tuning 3-Phase PMSM Sensorless Control Application Using MCAT Tool, Freescale Semiconductor（文档 AN4912）

Freescale 文档通过www.freescale.com在线提供。

13 缩略语

表 7 所列为本文档所用的缩略语。

表7. 缩略语

术语	意义
ADC	模数转换器
Back-EMF	反电动势—旋转电机产生的电压
FOC	磁场定向控制—矢量控制的同义词
PMSM	永磁同步电机
PWM	脉冲宽度调制
SVM	空间矢量调制—用于产生 PWM 输出信号的算法

14 修订历史

修订版 0 是本文档的初始版本。

联系我们：

主页：
freescale.com

Web支持：
freescale.com/support

文档编号：AN4935
修订版0
2014年5月