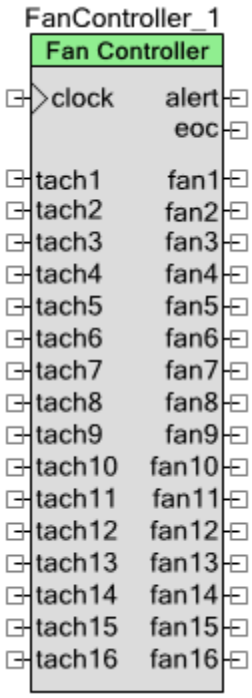
风扇控制器

3.10



特性

- PSoC 3/PSoC 5LP 器件支持多达 16 个 PWM 控制的 4 线无刷直流电风扇，PSoC 4 支持多达 6 个风扇
- 独立或分组的脉冲宽度调制器输出，带转速计输入
- 支持 25 kHz、50 kHz 或用户指定的脉冲宽度调制器频率
- 支持高达 25,000 RPM 的风扇速度
- 支持 4 极和 6 极电机
- 支持所有风扇上的风扇停顿/电机锁定检测
- 对于 PSoC 3/PSoC 5LP, 支持由固件控制的或由硬件控制的风扇速度调节
- 对于 PSoC 4, 支持由固件控制的风扇速度调节
- 用于风扇故障报告的可自定义的警报引脚

概述

风扇控制器组件有助于设计员快速、轻松地使用 PSoC 开发风扇控制器解决方案。该组件是一个系统级的解决方案，并且封装了所有必要的硬件模块，包括 PWM（或用于 PSoC 4 的 TCPWM）、转速计输入捕获定时器、控制寄存器、状态寄存器和 DMA 控制器（或用于 PSoC 4 的 ISR），以便减少开发时间和精力。

通过图形用户接口可以自定义该组件，从而使设计员能够输入风扇机电参数，如占空比到 RPM 的映射和物理风扇分组组织。通过同一个用户接口可以配置包含 PWM 频率和分辨率的性能参数以及开环或闭环控制方法。一旦输入了系统参数，该组件将提供最佳的实现方法以节省 PSoC 内的资源，从而使能其他热管理和系统管理功能的整合。另外，该组件还提供了各易于使用的 API，允许固件开发人员快速地启动和运行风扇控制功能。

注意：采用风扇控制器组件的设计不得使用 PSoC 的低功耗睡眠模式或休眠模式。进入这些模式会阻止风扇控制器组件对风扇进行控制和监控。

何时使用风扇控制器

在需要驱动和监控基于 PWM 的 4 线直流散热风扇的所有热管理应用中，都能使用风扇控制器组件。如果应用需要超过 16 个风扇，风扇控制器组件可以进行多次实例化（添加多个风扇控制器）。同样，如果应用中的风扇被分成各个组，设计人员可以选择每组实例化一个风扇控制器组件或实例化一个可处理所有分组的组件。

输入/输出接口

本节介绍了风扇控制器的各种输入和输出接口。I/O 列表中的星号（*）表示该 I/O 可能隐藏或可见的（取决于该 I/O 的描述部分中所列的条件的定义）。

时钟 — 输入*

用于风扇控制 PWM 的用户定义时钟源的输入。仅当在组件定制器中选中 External Clock（外部时钟）项时，该输入才显示。

tach1-16 — 输入 *

从每个风扇发出的转速计信号，用于使风扇控制器能够测量风扇旋转速度。设计该组件与 4 极直流风扇一起使用，它们在其转速计输出上每次旋转生成 2 个高低脉冲序列；或者与 6 极直流风扇一起使用， 它们将生成 3 个高低脉冲序列。tach2..16 输入是可选项。

注意：对于 PSoC 4 器件，因数字资源受限制，所以最多只能使用 6 个转速计输入。

fan1-16 — 输出*

可变占空比的脉冲宽度调制器输出，用于控制风扇速度。如果已使能风扇分组，这些输出终端将被 bank1..8 输出替换。

注意：对于自动硬件（UDB）模式，风扇输出（和关联的转速计输入）的最大数量仅限于 12，以尽量减少数字资源的占用。

注意：对于 PSoC 4 器件，因数字资源受限制，所以最多只能使用 4 个风扇输出。

bank1-8 — 输出*

可变占空比的脉冲宽度调制器输出，用于控制风扇组的速度。这些输出仅在使能了分组时才显示。

alert（报警） — 输出

当检测到了风扇故障（如果已使能）时，将设置高电平有效输出终端。风扇故障可能是风扇停顿/电机锁定，也可能是速度调节故障。该信号与“eoc”信号同步，并且在“eoc”脉冲后会立即将“alert”信号设置为高电平。该信号是一个“粘滞”信号。也就是说，当检测到故障条件时，该信号将被设置为高电平，并一直保持该状态，直到固件中的 GetAlertSource() API 被调用为止。如果没有找到并纠正故障条件，则在“eoc”信号脉冲之后将重新设置“alert”信号。如果出现风扇停顿故障，则在下一个“eoc”脉冲后会立即设置“alert”信号。出现速度调节故障时，会在连续发生一行中 16 个故障条件后生成警报信号。

eoc（周期末）— 输出

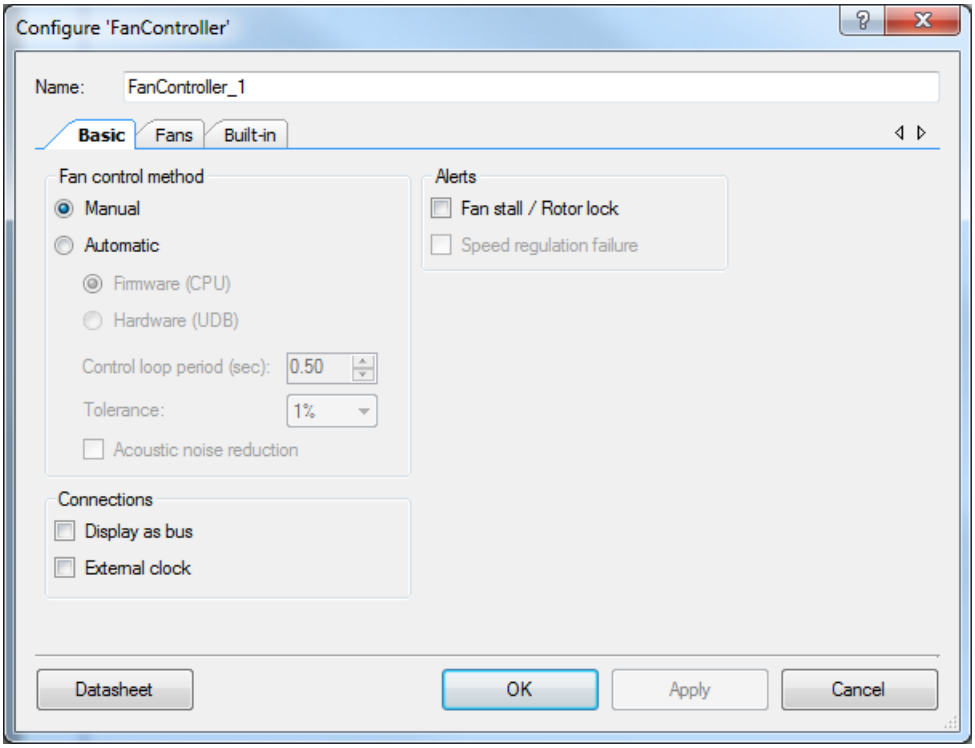
每次转速计模块测量了系统中所有风扇的速度后，End-of-Cycle（周期末）输出将处于高脉冲。它可以通过连接终端至状态寄存器组件或中断组件的方式，使固件算法与风扇控制器硬件同步。

Component Parameters （组件参数）

将风扇控制器拖入您的设计中，双击它打开 Configure（配置）对话框。

Basic（基本）选项卡

该选项卡用于配置该组件的基本工作参数。



该选项卡包含下列参数：

Fan control method（风扇控制方法）— Manual/Automatic（手动/自动）

该参数确定如何控制风扇速度。可用选项包括：

- Manual（手动）
- Automatic（自动）
  - Firmware (CPU)固件（CPU）
  - Hardware (UDB)硬件（UDB）

Manual（手动）

Manual（手动）速度控制是一个开环风扇控制模式，表示复杂性最低的实现。在该模式下，可以使用 SetDutyCycle() API 设置 PWM 输出占空比，而 无需使用或考虑实际的风扇速度。 因为在假定条件下风扇是以预定速度运行的。但仍会检测并报告风扇故障（停顿或锁定电机）。

在下面的情况中，应该设置手动速度控制：

1. 需要在 PSoC 内部的固件中实现复杂的或自定义的风扇控制算法。
2. 外部主机控制器负责管理风扇速度算法，而风扇控制器组件只是作为风扇的硬件接口使用的。
3. 多个风扇被分为不同的组，并共用一个 PWM 驱动信号。

Automatic Firmware (CPU)（自动固件（CPU））

Automatic Firmware (CPU)模式采用组件附带的固件 PID 算法来控制速度调节。当选中该模式时，在组件的“Configure”（配置）对话框中将出现一个额外的选项卡，即 **PID Control**。通过该选项卡，可以输入各个 PID 算法参数。根据这些参数，在 ISR 中执行的 PID 算法会分析预定和实际风扇的 RPM，并为当前受控制的风扇设置合适的占空比。

通过该模式，并使用在固件中运行的优化 PID 控制循环，可以独立控制多个风扇。

Automatic Hardware (UDB)（自动硬件（UDB））

Automatic Hardware (UDB)模式表示 PSoC 内的硬件模块自动控制风扇的速度调节，而无需 CPU 的任何干预。固件为每个风扇设置所需要的速度，同时，硬件会自动调节 PWM 占空比，以在指定的容差范围内获取并维持所需的速度。

如要通过最少的固件开发控制多个风扇，要选择自动硬件（UDB）设置。

注意：对于 PSoC 4 器件，由于它的 UDB 资源受限制，所以 Automatic Hardware (UDB)选项被禁用。

Control loop period（控制循环周期）

在自动控制风扇模式下，该参数用于控制自动硬件或固件控制循环（闭环）的动态响应时间。该参数控制着内部算法间隔多久调节一次每个风扇的 PWM 占空比。该参数实现了硬件控制逻辑的微调，以符合选定风扇的机电特性。该参数的有效范围为 0 到 2.55（单位为秒）。默认值为 0.5。

Tolerance（容差）

该选项只在自动硬件（UDB）控制风扇模式中显示。在指定所需要的风扇速度目标时，该参数设置可接受的容差。该容差被指定为与所需的速度设置相关的百分比。该参数实现了硬件控制逻辑的微调，以符合选定风扇的机电特性。

该参数的有效范围为 1 到 10%。默认设置为 1%。如果在 **Fan Controller Fans**（风扇控制器风扇）选项卡中选择了 8 位 PWM 分辨率，推荐使用 5%的 Tolerance（容差）参数。

Acoustic Noise Reduction（吸声降噪）

在自动硬件（UDB）控制风扇模式下，该参数通过限制速度的正向变化率来限制风扇的可听噪音。如果已使能，且固件请求提高所需的风扇速度，应用于风扇的 PWM 占比空将逐渐提高到新的设置，而不是应用突然性的改变。这消除了由于突然性的速度增加而导致的嘈杂的风扇呼呼声。默认本选项被选中。由于该选项是 PID 算法的内置设置，所以该选项在自动固件（CPU）模式中被禁用。

各个连接

- **显示为总线—** 如果已选中，转速计输入和风扇/分组输出将显示为总线。否则，它们将显示为独立的终端。
- **外部时钟 —** 如果已选中，您可将外部时钟连接至组件时钟输入。如果未选中，将使用内部时钟源。

Alerts（警报）

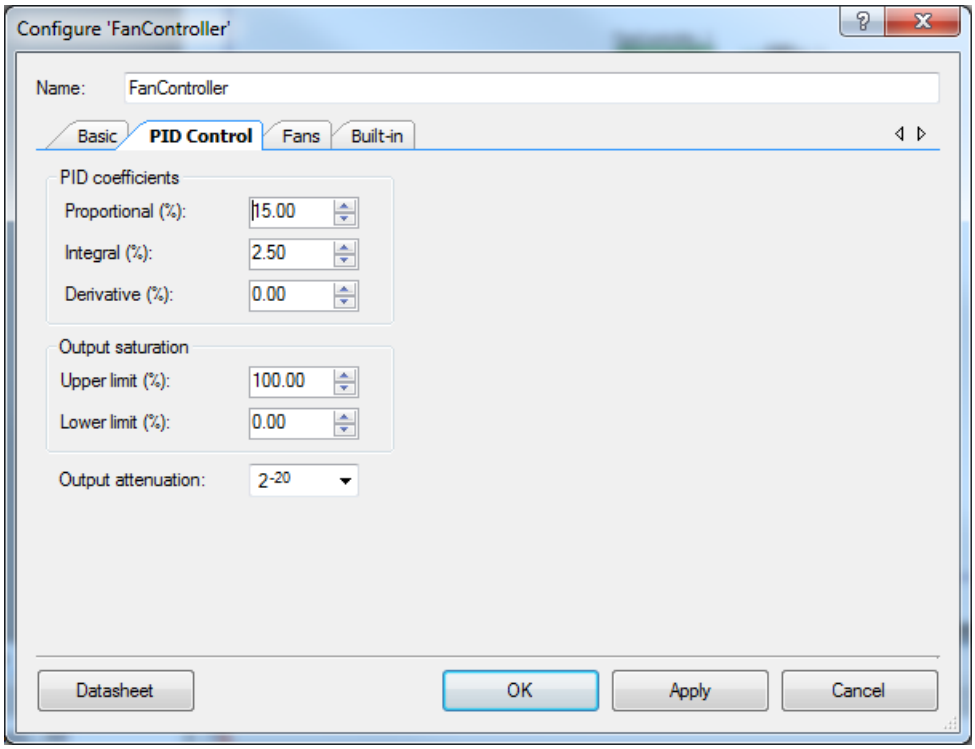
- **Fan stall / Rotor lock**（风扇停顿/电机锁定） **—** 在风扇因机械性阻塞而延迟或停止时，可通过配置风扇控制器来生成高电平有效警报信号。当风扇速度低于 500 转每分时，也会生成风扇停顿/电机锁定的警报信号。默认本选项被选中。
- **Speed regulation failure**（速度调节失败） **—** 当自动控制循环无法达到所需要的速度时，可通过配置风扇控制器，以在自动控制风扇模式中生成高电平有效警报信号。默认本选项未选中。

  如果自动控制算法不能使风扇转速达到所需值，那么会发生速度调节故障。具体是，如果已经将 PWM 驱动风扇的占空比设置为 100%，但实际速度仍低于所需的速度，则会发生这

种故障。如果 PWM 已设置为 0%但实际速度仍高于所需速度，也会发生故障。为了避免发生失败警报，在 16 个连续的速度测量周期内发生故障条件，将会生成速度调节故障。

PID Control（PID 控制）选项卡

该选项卡用于配置固件 CPU 控制模式的系数。



该选项卡包含下列参数：

PID Coefficients（PID 系数）

在 PID 控制算法中使用了下面三个增益参数：**比例增益（K_P）， 积分增益（K_I）以及微分增益（K_D）**。 每个参数的比率范围为 0.00%到 100.00%。这些参数的默认设置为大多数系统提供了低速且稳定的性能（即低带宽 PI 控制）。启动时，这些系数适用于该配置中的每个风扇，并可以通过 FanController_SetPID() API 对其进行更改。此外，可对每个风扇的这些值进行独立更改。

有关PID系数的更过信息，请查看本数据手册中的PID算法部分。

Output saturation（输出饱和）

- **Upper limit**（上限） — 该参数定义积分的初始上限。其比率范围为 0.00%到 100.00%。该参数可用于设置风扇的最大系数；也可以用来阻止无限控制系统中常见的积分饱和。启动时，该参数值适用于该配置中的每个风扇，并可以通过 FanController_SetSaturation() API 对其进行更改。此外，可对每个风扇的参数值进行独立更改。

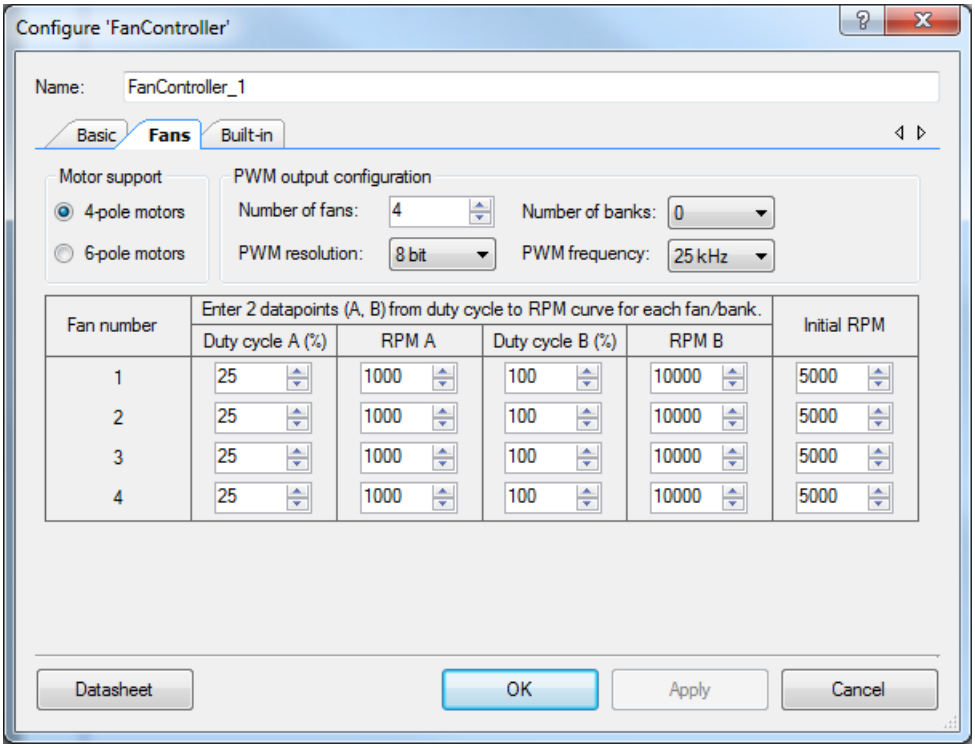
- **Lower limit**（下限） — 该参数定义积分的初始下限。其比率范围为 0.00%到 100.00%。该参数可用于设置风扇的最小系数；也可以用来阻止无限控制系统中常见的积分饱和。启动时，该参数值适用于该配置中的每个风扇，并可以通过 FanController_SetSaturation() API 对其进行更改。该参数设置的值要始终低于它的饱和上限值。此外，可对每个风扇的参数值进行独立更改。

Output attenuation（输出衰减）

这是一个二的幂次的指数，其范围值为2^{-8}到2^{-23}。默认增益设置为大多数系统提供了低速且稳定的性能（即低带宽PI控制）。

Fans（风扇）选项卡

该选项卡用于配置风扇特定的参数。



该选项卡包含下列参数：

Motor support（电机支持）

该参数指定每转在风扇的转速计输出上显示的高-低脉冲的数量。4 极选项表示风扇每一转会有两个高脉冲和两个低脉冲。6 极选项表示风扇每一转会有三个高脉冲和三个低脉冲。

PWM 输出配置

- **Number of fans**（风扇数量）— 该参数表示系统中所使用的风扇数量。其有效范围为 1 到 16。默认值为 4。

  注意对于自动硬件（UDB）模式，风扇的数量限于 12。在 PSoC 4 器件的自动固件（CPU）模式中，风扇的最大数量为 4；在手动模式中，该数量为 6（使用分组功能）。

- **Number of banks**（分组的数量）— 该参数只在固件控制模式中显示。它指定系统中风扇分组的数量。假设当风扇被分成各个组时，每一组都有相同的风扇数量。因此，风扇分组的数量必须是风扇数量的约数。0 值表示风扇未进行分组。对于分组操作，该参数的有效值为 1 到（风扇数量/2）。默认设置为 0。

- **PWM resolution**（PWM 分辨率）— 参数为旋转速度控制指定了用于驱动风扇的调制 PWM 信号的占空比分辨率。其有效选项为 8 位或 10 位。默认设置为 8-bit（8 位）。

- **PWM frequency**（PWM 频率）— 该参数指定用于驱动风扇的调制 PWM 信号的频率。当使用了内部时钟时，该参数的有效配置为 25 kHz 到 50 kHz。默认设置为 25 kHz。当使用了外部时钟时，该参数显示为灰色，因为 PWM 频率取决于输入时钟源。更多详细信息，请参考功能说明部分。

风扇规范

- **Duty cycle A (%)、RPM A**（占空比 A（%）、RPM A） — 这些参数为选定的风扇或风扇分组在占空比到 RPM 的传输功能上指定一个数据点。RPM A 参数指定了当 PWM 将风扇驱动到占空比 Duty A（%）时该风扇正常运行的速度。可从风扇制造商的数据手册中获得有关该速度的信息。请注意，即使 Duty A (%)被设置为非零值，风扇控制器组件仍会使 PWM 占空比下降至 0%。
  - 占空比 A （%）参数的有效范围为 0 至 99。默认值为 25。
  - RPM A 参数的有效范围为 500 到 24,999。默认值为 1,000。

- **Duty cycle B（%）、RPMB**（占空比 B（%）、RPM B）— 这些参数为选定的风扇或风扇分组在占空比到 RPM 的传输功能上一起指定第二个数据点。RPM B 参数指定当风扇由 PWM 使用占空比 Duty B （%）驱动时风扇正常运行时的速度。可从风扇制造商的数据手册中获得有关该速度的信息。应注意，即使 Duty B（%）设置低于 100%，风扇控制器组件仍可能将 PWM 占空比驱动到 100%。
  - Duty B（%）参数的有效范围为 1 至 100。默认值为 100。

  - RPM B 参数的有效范围为 501 到 25,000。默认值为 10,000。

- **Initial RPM**（初始 RPM）— 该参数指定单个风扇的初始 RPM。初始 RPM 的值将被转换为占空比，并被设置为单个风扇的初始占空比。可将 Initial RPM 参数设置为低于 RPM A 参数的值。

时钟选择

该组件使用时钟树资源进行其操作。这些时钟分别为：总线时钟（不适用于 PSoC 4）、转速计时钟（500 kHz）以及 PWM 时钟（根据配置，其频率可为 6、12 或 24 MHz）。该组件有一个连接外部时钟源而非 PWM 时钟的选项，以生成所需的 PWM 输出频率。

注意：对于 PSoC 4 器件中带 10 位 PWM 分辨率的组件，由于时钟的限制，需要将 IMO 的频率设置为 48 MHz。

应用编程接口

应用编程接口（API）子程序允许您使用固件与组件交互。下表列出并说明了每个函数的接口。以下各节将更加详细地介绍每个函数。

默认情况下，PSoC Creator 将实例名称 FanController_1 分配给指定设计中的第一个组件实例。您可以将其重新命名为遵循标识符语法规则的任何唯一值。实例名称会成为每个全局函数名称、变量以及符号常量的前缀。出于可读性考虑，下表中使用的实例名称为 FanController。

函数

函数	说明
FanController_Start()	启动该组件
FanController_Stop()	停止该组件并禁用硬件模块
FanController_Init()	初始化该组件
FanController_Enable()	使能该组件中的硬件模块
FanController_EnableAlert()	使能该组件中的警报
FanController_DisableAlert()	禁用该组件中的警报
FanController_SetAlertMode()	配置警报源
FanController_GetAlertMode()	返回当前使能的警报源
FanController_SetAlertMask()	使能每个风扇的警报掩码
FanController_GetAlertMask()	返回每个风扇的警报掩码状态

函数	说明
FanController_GetAlertSource()	返回待处理的警报源
FanController_GetFanStallStatus()	返回表示每个风扇的停顿状态的位掩码
FanController_GetFanSpeedStatus()	返回表示每个风扇在硬件控制模式中的速度调节状态的位掩码
FanController_SetDutyCycle()	为指定的风扇或风扇分组设置PWM占空比
FanController_GetDutyCycle()	返回指定风扇或风扇分组的PWM占空比
FanController_SetDesiredSpeed()	设置指定风扇在硬件控制模式中所需要的风扇速度
FanController_GetDesiredSpeed()	返回指定风扇在硬件控制模式中所需要的风扇速度
FanController_GetActualSpeed()	返回指定风扇的实际速度
FanController_OverrideAutomaticControl()	使能用于覆盖自动风扇控制的固件
FanController_SetSaturation()	更改PID控制器的输出饱和
FanController_SetPID()	更改控制风扇的PID控制器系数

全局变量

变量	说明
FanController_initVar	initVar变量用于说明该组件的初始配置。该变量前面加有组件名称。该变量被初始化为0，并在第一次调用FanController_Start()时设置为1。这实现了组件初始化，而无需重新初始化FanController_Start()子程序中的所有后续调用。 如要重新初始化此组件，应首先调用FanController_Stop()子程序，然后再调用FanController_Init()和FanController_Enable()。

void FanController_Start(void)

说明： 使能组件。如果之前未初始化组件，请调用Init() API。随后调用Enable() API。

参数： 无

返回值： 无

其他影响： 无

void FanController_Stop(void)

说明： 禁用组件。所有的PWM输出都会被驱使为100%占空比，以确保在该组件不运行时持续散热。

参数： 无

返回值： 无

其他影响： 警报引脚已解除。

void FanController_Init(void)

说明： 初始化组件。

参数： 无

返回值： 无

其他影响： 无

void FanController_Enable(void)

说明： 使能该组件中的硬件模块。

参数： 无

返回值： 无

其他影响： 无

void FanController_EnableAlert(void)

说明： 使能警报信号的生成。特指使用FanController_SetAlertMode()和FanController_SetAlertMask() API配置所需使能的警报源。

参数： 无

返回值： 无

其他影响： 无

void FanController_DisableAlert(void)

说明： 禁用警报信号的生成。

参数： 无

返回值： 无

其他影响： 警报引脚已解除。

void FanController_SetAlertMode(uint8 alertMode)

说明： 配置该组件的警报源。可用的警报源有两个：1) 风扇停顿或电机锁定； 2) 硬件控制模式的速度调节失败。

参数： uint8 alertMode

位字段	使能警报源
FanController_STALL_ALERT	1 = 使能风扇停顿/电机锁定警报
FanController_SPEED_ALERT	1 = 使能自动控制速度调节失败警报

返回值： 无

其他影响： 无

uint8 FanController_GetAlertMode(void)

说明： 返回已使能的警报源。

参数： 无

返回值： uint8 alertMode

位字段	使能警报源
FanController_STALL_ALERT	1 = 使能风扇停顿/电机锁定警报
FanController_SPEED_ALERT	1 = 使能闭环速度调节失败警报

其他影响： 无

void FanController_SetAlertMask(uint16 alertMask)

说明： 通过掩码使能或禁用来自每个风扇的警报。掩码同时应用于风扇停顿警报和速度调节失败警报。

参数： uint16 alertMask

位字段	使能警报源
位0	1 = 使能Fan1的警报
位1	1 = 使能Fan2的警报
…	…
位15	1 = 使能Fan16的警报

返回值： 无

其他影响： 无

uint16 FanController_GetAlertMask(void)

说明： 返回每个风扇的警报掩码状态。掩码同时应用于风扇停顿警报和速度调节失败警报。

参数： 无

返回值： uint16 alertMask

位字段	使能警报源
位0	1 = 使能Fan1的警报
位1	1 = 使能Fan2的警报
…	…
位15	1 = 使能Fan16的警报

其他影响： 无

uint8 FanController_GetAlertSource(void)

说明： 返回该组件待处理的警报源。该API可用于轮询该组件的警报状态。或者，如果警报引脚用于生成PSoC的CPU内核中断，中断服务子程序可使用该API确定警报源。在上述任何情况下，如果该API返回了非零值，FanController_GetFanStallStatus()和FanController_GetFanSpeedStatus() API可提供有关哪个（些）风扇出现了故障的详细信息。

参数： uint8 alertMode

位字段	待处理警报
FanController_STALL_ALERT	1 = 风扇停顿/电机锁定警报正等待处理
FanController_SPEED_ALERT	1 = 闭环速度调节失败警报正等待处理

返回值： 无

其他影响： 调用该API会取消确认警报引脚。如果有任何警报被挂起，警报引脚将在下一个周期末（eoc）脉冲后被立即取消确认。

uint16 FanController_GetFanStallStatus(void)

说明： 返回所有风扇的停止/电机锁定状态。

参数： 无

返回值： uint16 stallStatus

位字段	状态
位0	Fan1停止状态（1 = 停止，0 = 正常）
位1	Fan2停止状态
…	…
位15	Fan16停止状态

其他影响： 调用该API将清除所有挂起的风扇停止警报。

uint16 FanController_GetFanSpeedStatus(void)

说明： 返回所有风扇的硬件风扇控制模式速度调节状态。在下面两种情况下，会发生速度调节故障：1) 如果所需的风扇速度超过了当前的实际风扇速度，但风扇的占空比已被设置为100%；2) 如果所需的风扇速度低于当前的实际风扇速度，但风扇的占空比已被设置为0%。

参数： 无

返回值： uint16 speedStatus

位字段	状态
位0	Fan1速度调节状态（1 = 失败，0 = 正常）
位1	Fan2速度调节状态
…	…
位15	Fan16速度调节状态

其他影响： 调用该API将清除所有挂起的速度调节警报。

void FanController_SetDutyCycle(uint8 fanOrBankNumber, uint16 dutyCycle)

说明： 设置指定风扇或风扇分组的脉冲宽度调制器占空比（单位为万分比）。在硬件风扇控制模式中，如果需要手动占空比控制，在调用该API之前首先调用FanController_OverrideHardwareControl()。

参数： uint8 fanOrBankNumber

风扇或风扇分组。有效范围为1…16，但不得超过系统中的风扇或分组的数量。

uint16 dutyCycle

占空比（单位为万分比）。例如，50%的占空比 = 5000。有效范围为0..10000。

返回值： 无

其他影响： 无

uint16 FanController_GetDutyCycle(uint8 fanOrBankNumber)

说明： 返回指定风扇或风扇分组的当前脉冲宽度调制器占空比（单位为万分比）。

参数： uint8 fanOrBankNumber

风扇或风扇分组。有效范围为1…16，但不得超过系统中的风扇或分组的数量。

返回值： 占空比（单位为万分比）。例如，50%的占空比 = 5000。

其他影响： 无

void FanController_SetDesiredSpeed(uint8 fanNumber, uint16 rpm)

说明： 设置指定风扇需要的速度（单位为每分钟转数（RPM））。在硬件风扇控制模式中，RPM参数被传送至控制循环硬件，作为调节的新目标风扇速度。在固件风扇控制模式中，根据定制器的Fans选项卡中输入的风扇参数，将RPM参数转换为被写入到适当的脉冲宽度调制器中的占空比。这向固件提供了初始粗粒度级速度控制的方法。可使用FanController_SetDutyCycle() API达成高精度固件速度控制。

参数： uint8 fanNumber

风扇或风扇分组。有效范围为1…16，但不得超过系统中风扇的数量。

uint16 rpm

有效范围为500..25,000，但不得超过风扇能够运行的最大RPM。否则会导致速度调节失败。

返回值： 无

其他影响： 无

uint16 FanController_GetDesiredSpeed(uint8 fanNumber)

说明： 返回选定风扇当前需要的速度。

参数： uint8 fanNumber
风扇或风扇分组。有效范围为1…16，但不得超过系统中风扇的数量。

返回值： RPM中选定风扇当前需要的速度。

其他影响： 无

uint16 FanController_GetActualSpeed(uint8 fanNumber)

说明： 返回选定风扇当前的实际速度。

参数： uint8 fanNumber
风扇数量。有效范围为1…16，但不得超过系统中风扇的数量。

返回值： RPM中选定风扇当前的实际速度。

其他影响： 无

void FanController_OverrideAutomaticControl(uint8 override)

说明： 允许固件在硬件风扇控制模式中控制风扇。注意，在固件风扇控制模式下，无法调用该API。

参数： uint8 override
0 = 硬件负责风扇的控制
1 = 固件负责风扇的控制
有效范围为0..1。默认设置为0

返回值： 无

其他影响： 无

void FanController_SetSaturation(uint8 fanNum, uint16 satH, uint16 satL)

说明： 更改PID控制器的输出饱和。这样将限制风扇的输出PWM并防止各积分饱和。

参数： uint8 fanNum

风扇数量。有效范围为1…16，但不得超过系统中风扇的数量。

uint16 satH

饱和的高阈值。有效范围为0至65535。该值为0表示占空比为0%。该值为65535则表示占空比为100%。

uint16 satL

饱和的低阈值。有效范围为0至65535。该值为0表示占空比为0%。该值为65535表示占空比为100%。

返回值： 无

其他影响： 无

void FanController_SetPID (uint8 fanNum, uint16 kp, uint16 ki, uint16 kd)

说明： 更改控制风扇的PID控制器系数。这些系数是与增益成正比的整数。

参数： uint8 fanNum：风扇数量。有效范围为1…16，但不得超过系统中风扇的数量。

uint16 kp：比例增益。有效范围为0至65535。该值为0表示增益为0%。该值为65535则表示增益为100%。

uint16 ki：积分增益。有效范围为0至65535。该值为0表示增益为0%。该值为65535则表示增益为100%。

uint16 kd：微分增益。有效范围为0至65535。该值为0表示增益为0%。该值为65535则表示增益为100%。

返回值： 无

其他影响： 无

MISRA 合规性

本节介绍了 MISRA-C:2004 规范以及本组件的偏差情况。有两种偏差类型，如下定义：

- 项目偏差 — 适用于所有 PSoC Creator 组件的偏差
- 特定偏差 — 仅适用于该组件的偏差

本节介绍了有关组件特定偏差的信息。*系统参考指南*的“MISRA 合规性”章节中介绍了项目偏差以及有关 MISRA 合规性验证环境的信息。

风扇控制器组件具有以下特定偏差：

MISRA-C：2004 规则	规则类别（必须（R）/ 建议（A））	规则说明	偏差说明
10.3	必须	不能对不同对象指针类型进行转换。	使用自动固件模式时，PID算法将有符号整数变量更改为无符号整数变量。
13.7	必须	不允许结果不变的布尔运算。	根据组件配置，该组件可以进行条件检查，这些条件是不变的。
19.7	建议	函数应比类函数宏优先使用。	添加了类函数宏 FanController_OverrideHardwareControl()，用于支持现有的设计。这是因为函数 FanController_OverrideHardwareControl() 被重新命名为 FanController_OverrideAutomaticControl()
21.1	必须	应至少通过使用以下各项中的一个，以确保最大程度地减少运行时发生的故障： a)静态分析工具/技术 b)动态分析工具/技术 c)用于处理运行过程中发生故障的检查的明确编码	根据组件配置，该组件可以进行条件检查，并且这些条件是不变的。因此将冗余代码中的某些部分。

该组件具有以下嵌入式组件：DMA，ISR。MISRA 合规性与特定偏差的相关信息，请参见相应组件数据手册。

固件源代码示例

在“Find Example Project”对话框中，PSoC Creator 提供了大量的示例项目，包括原理图和代码。要获取组件特定的示例，请右击组件并打开“Find Example Project”对话框来查看组件实例。要查看通用示例，请打开‘Start Page’或 **File** 菜单中的对话框。根据要求，可以通过使用对话框中的 **Filter Options** 选项来限定可选的项目列表。

更多有关信息，请参考 PSoC Creator 帮助中的“Find Example Project”（查找示例项目）主题。

中断服务子程序

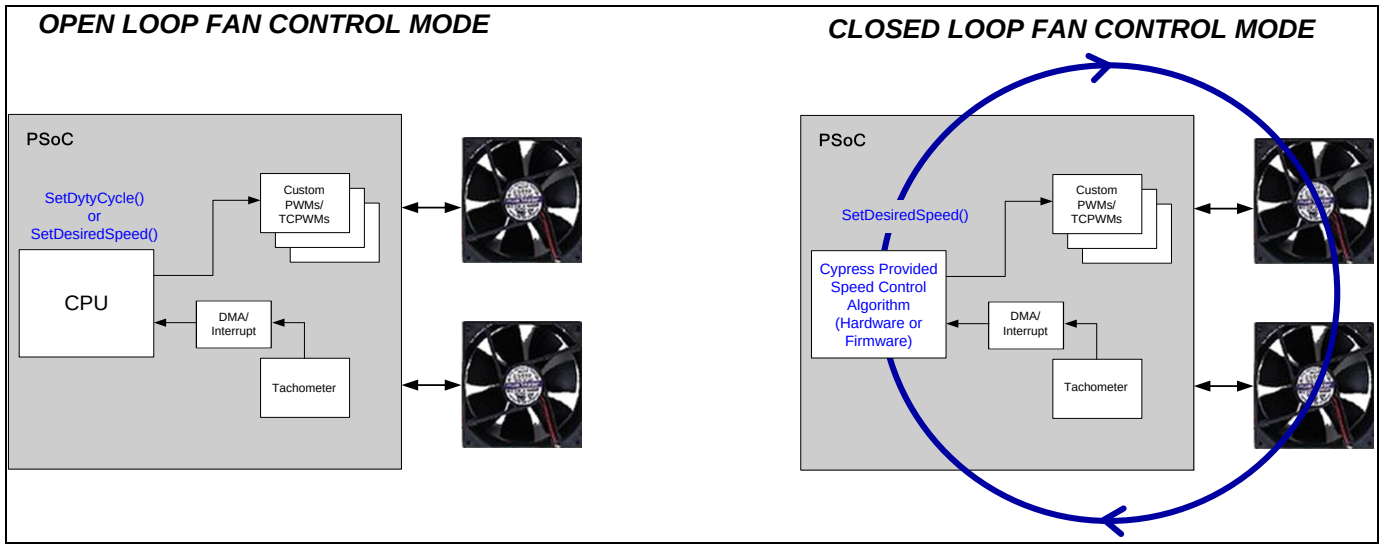
根据配置模式和被使用的器件，风扇控制器组件可以使用两个中断。当配置为自动固件（CPU）模式时，该组件使用 *FanController_PID_ISR* 中断。该中断实现 PID 控制算法。在 PSoC4 中使用该组件时，*FanController_DataSend* 中断将被使用。通过该中断可以将所测量的 RPM 速度从转速计传输到 RAM。

注意：除了风扇控制器中断外，如果 PSoC 4 设计还使用了其他中断，*FanController_DataSend* 中断的优先级将最高。这是因为 *FanController_DataSend* 必须在小时间窗口期间读取当前有效的风扇地址，并将实际速度的读取值置位在合适的 RAM 地址。

功能说明

框图和配置

下图所示的原理图显示的是该组件的两种基本工作模式的高级框图：1)开环控制模式（手动模式），2)闭环控制模式（自动模式）。



手动（开环）控制模式很简单，具体情况在组件参数章节中介绍。

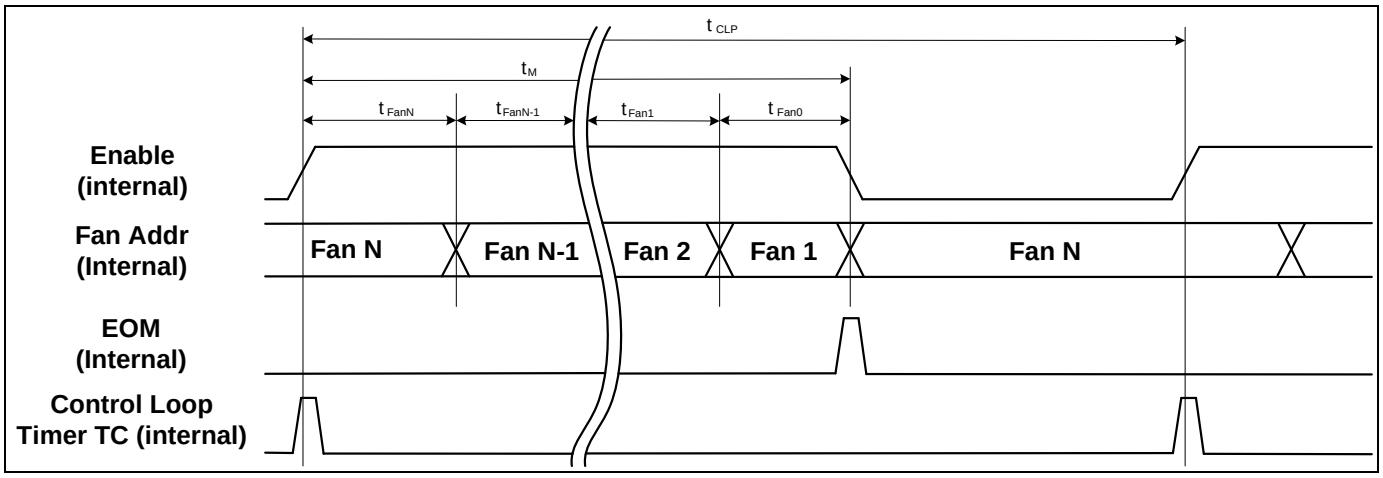
自动（闭环）控制模式

在自动控制模式下，控制算法被隐藏，但组件提供了一组参数，用于配置已选定的算法。虽然该算法几乎不需要任何固件开发，但它需要组件操作的一些基本信息，以对其进行合适配置。

控制循环周期

控制循环周期在各风扇速度调整之间提供了一个可编程的延迟。该延迟的主要目的是为风扇提供时间以使其达到稳定状态，这样可以避免风扇振荡现象。对于自动硬件（UDB）模式，控制循环周期是可选的。当组件被配置用于多个风扇（例如 12 个或更多）时，对这些风扇进行测量将引入一个自然延迟，因此不需要控制循环周期。自动固件（CPU）模式需要控制循环周期（t_{CLP}）。该周期要始终大于测量全部风扇速度所需要的时间（t_M），在自动硬件（UDB）模式下，t_{CLP} 可以小于 t_M。

下面的时序图显示的是控制循环周期的生成。



在上面的框图内，当内部控制循环定时器的终端计数为高电平时，内部使能信号将为高电平。这样将使能信号开始对所有风扇进行速度测量序列。当内部测量结束（EOM）信号脉冲时，该序列将停止，并确认使能信号为低电平。

控制循环周期计算

在自动硬件（UDB）模式下，控制循环周期只有大于测量所有风扇的时间时才会起作用。在自动固件（CPU）模式下，必须保证 $t_{CLP} > t_M$。这是因为 PID ISR 在控制循环周期定时器的每个终端计数上被触发，以运行 PID 算法，而且此时全部风扇的所有测量实际速度值必须都可用。

根据下面的公式可以计算 t_M：

$$t_M = t_{Fan1} + t_{Fan2} + .. + t_{FanN} ; \quad (1)$$

其中 t_{Fan1} ... t_{FanN} 是测试 Fan1..。FanN 速度时需要的时间周期。 测量风扇速度需要的时间通过下面公式计算得出：

$$t_{Fan} = 1.75 * (60/RPM_{Fan_min}); \quad (2)$$

在该公式中，系数 1.75 表示硬件规定的风扇旋转为 1.75，用于测量速度。RPM_{Fan_min} 是特定风扇的最小速度。

例如，4 个风扇（RPM_{Fan_min} = 1000）的配置结果如下：

$t_M = 4 * 1.75 * (60/1000) = 0.42$ 秒

但 $t_{CLP} > t_M$，因此该配置的控制循环周期为 0.43 秒。

自定义时钟

该组件具有允许连接外部时钟的功能。时钟源的频率确定脉冲宽度调制器的输出频率，且输入时钟频率（f_{CLK}）与输出脉冲宽度调制器频率（f_{PWM}）之间的关系如以下公式所示：

$$f_{PWM} = f_{CLK} / P_{PWM}; \quad (3)$$

其中 P_{PWM} 是 PWM 周期。对于 8 位分辨率 P_{PWM} 为 240，对于 10 位分辨率， P_{PWM} 为 960。

PID 算法

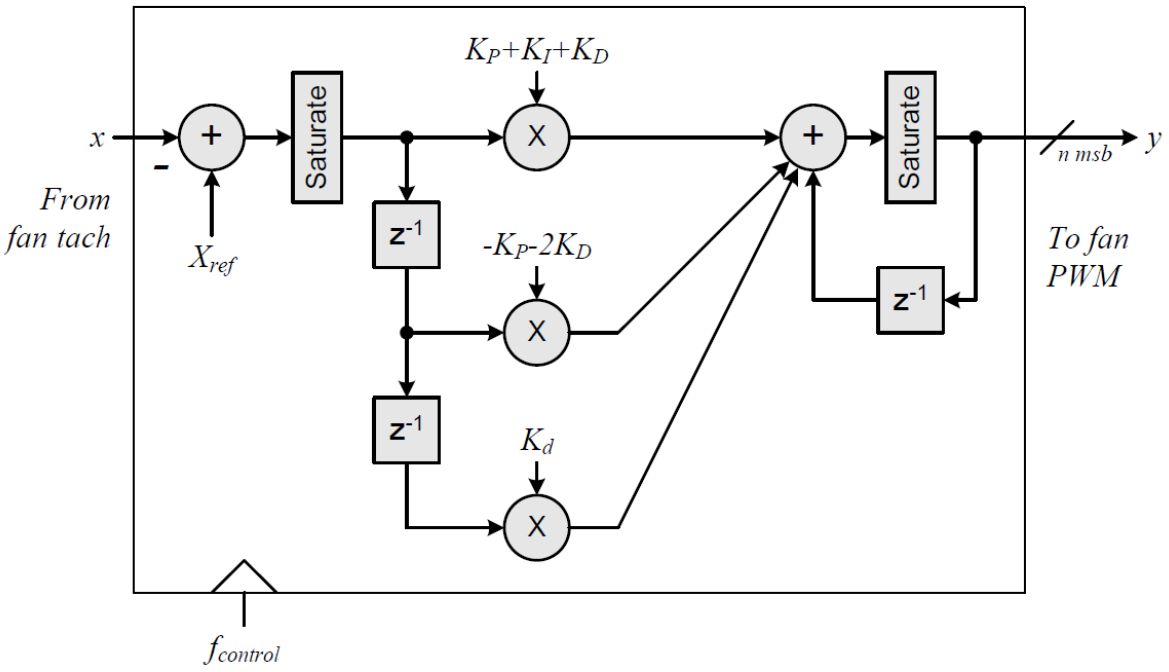
自动固件 CPU 控制模式使用比例-积分-微分（PID）算法来控制各个风扇。通过下面的公式可以确定该算法：

$$G_{PID}[z] = K_P + K_I * (1 / (1 - z^{-1})) + K_D * (1 - z^{-1}); \quad (4)$$

虽然可以在代码中直接实现公式 4，但该结果会在 PSoC 的处理过程和内存资源上生成极度滥用的代码。然而，可以将公式 4 代数简化为更有效的一种格式，具体如公式 5 所示：

$$G_{PID}[z] = ((K_P + K_I + K_D) + (- K_P + 2 * K_D) * z^{-1} + * (K_D) * z^{-2}) / (1 - z^{-1}); \quad (5)$$

该传输函数代表风扇控制器需要的 PID 算法。请注意，派生术语可能不需要用于大多数风扇控制应用，但在一个未知的风扇系统需要额外补偿的情况下，它被保持。下图显示的是公式 5 中传输函数的相应信号流框图。



为了帮助理解，后面的表格提供了 PID 控制算法的输入和输出简短摘要。

信号	方向	注意
PID_refference (X_{ref})	输入	控制调节的参考电压。它在风扇控制器中作为所需要的速度使用，单位为RPM。
PID_in (x)	输入	信号被控制，这时测量的风扇速度位于RPM内。
PID_error_saturation_high	输入	输入错误信号的饱和边界。这些值是硬编码为4096（高电平）和-4096（低电平）的内部常数。它们既可以用于计算宽输入错误，也符合控制器的计算限制。
PID_error_saturation_low		
PID_A1	输入	控制器系数，其中A1、A2和A3分别是KP、KI和KD的函数。在需要微调补偿以满足一个非常非线性的风扇响应情况下，通过FanController_SetPID()函数运行时，可以交换这些系数。然而，几乎在所有情况下，这些参数应该为常数。这些系数是限制为小于16位的带符号的整数。
PID_A2		
PID_A3		
PID_output_saturation_high	输入	需要将PID输出限制为被控制系统的动态范围，在这种情况下，PWM将驱动风扇。这些参数设置积分的上限和下限（防止积分饱和）。通过FanController_SetSaturation()函数运行时，用户可以交换这些参数。请注意，饱和是根据PID_POST_GAIN项的比例测量的。
PID_output_saturation_low		
PID_POST_GAIN (G_O)	输入	控制器的输出增益。用于计算输出饱和的高电平限制和低电平限制。该参数与输出衰减成反比。
PID_out (y)	输出	PID的输出，该输出将被发送到驱动风扇的PWM内。

通过公式 6、7 和 8 可以确定该算法使用的系数。

$$PID_A1 = (K_P + K_I + K_D) * 2^{12} ; \quad (6)$$

$$PID_A2 = -(K_P + 2 * K_D) * 2^{12} ; \quad (7)$$

$$PID_A3 = (K_D) * 2^{12} ; \quad (8)$$

注意：各公式所提供的系数结果均处于 12 位至 13 位的范围内，最大值不能超过 13 位。因此，各前馈项的合并结果不能在控制器集成阶段中引起溢出，从而不会产生任何小错误。

输出饱和的最大和最小饱和级由定制器设置。该定制器生成上限（YH）、下限（YL）、前部分中的 PWM 周期（PPWM）和输出增益（GO）。

$$PID_output_saturation_high = (Y_H * P_{PWM})/G_O; \quad (9)$$

$$PID_output_saturation_low = (Y_L * P_{PWM})/G_O; \quad (10)$$

寄存器

风扇控制器有多个控制寄存器和状态寄存器，它们被固件 API 使用以控制操作和监控状态。用户固件不能直接访问任何寄存器。

组件调试窗口

风扇控制器组件支持 PSoC Creator 组件调试窗口。在风扇控制器组件调试窗口中显示了以下各寄存器。

FanController_GLBL_CTRL

这是组件的全局控制寄存器。

FanController_ALERT_STATUS

这是组件的警报状态寄存器。

FanController_STALL_STATUS_LSB

该寄存器保持风扇 1-8 的停止警报状态。

FanController_MSB_STALL_STATUS_MSB

该寄存器保持风扇 9-16 的停止警报状态。

FanController_ALRT_MASK_LSB

该寄存器保持风扇 1-8 的速度警报状态。

FanController_MSB_ALRT_MASK_MSB

该寄存器保持风扇 9-16 的停止速度状态。

资源

风扇控制器组件放置在整个 UDB 阵列中。下表显示的是组件利用的 UDB 资源。

PSoC 3/PSoC 5LP

配置[1][2]	资源类型					
	数据路径单元	宏单元[3]	状态单元	控制单元	DMA通道	中断
手动模式，4个风扇	3 (7)	32	3	3	1	–
手动模式，8个风扇	7 (11)	40	3	3	1	–
手动模式，12个风扇	9 (15)	49	4	4	1	–
手动模式，16个风扇	11 (19)	59	4	4	1	–
自动固件模式，4个风扇	4 (8)	37	3	3	1	–
自动固件模式，8个风扇	8 (12)	45	3	3	1	–
自动固件模式，12个风扇	10 (16)	54	4	4	1	–
自动固件模式，16个风扇	12 (20)	64	4	4	1	–
自动硬件模式，4个风扇	7 (11)	61	4	7	2	–
自动硬件模式，8个风扇	11 (19)	97	4	11	2	–
自动硬件模式，12个风扇	15	135	6	16	2	–

PSoC 4

配置	资源类型						
	Datapath单元	宏单元	状态单元	控制单元	DMA通道	TCPWM模块	中断
手动模式，4个风扇	3	19	2	3	–	4	1
自动固件模式，4个风扇	4	25	2	3	–	4	2

1 说明 8 位分辨率资源的使用的表。由 10 位配置占用的资源，如果它们有差异，则它们将被显示在括号内。

2 对于硬件模式，不包括由控制循环周期功能使用的资源。

3 这些宏单元的数量是针对 PSoC 3 器件来说的。在 PSoC 5LP 中，组件使用 3 个宏单元（小于在 PSoC 3 内）。这是因为同时对 CPU 和 DMA 进行多字节访问的问题已得到修正。

API 存储器占用情况

根据编译器、器件、所使用的 API 数量以及组件的配置不同，组件的内存使用量也不一样。下表提供了组件配置中所有 API 占用的存储器大小。

下表中的存储器大小是在将相应编译器设置为 Release（释放）模式并且优化选项为 Size 的情况下测得的。对于特定的设计，可分析编译器生成的映射文件以确定存储器的使用情况。

配置[4][5]	PSoC 3 （Keil_PK51）		PSoC 4（GCC）		PSoC 5LP（GCC）	
	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节
手动模式，4 个风扇	1689 (1718)	82	1440	65	792 (816)	70
手动模式，8 个风扇	1705 (1734)	162	N/A	N/A	804 (820)	138
手动模式，12 个风扇	1738 (1767)	242	N/A	N/A	872 (896)	206
手动模式，16 个风扇	1754 (1783)	322	N/A	N/A	900 (932)	274
自动固件模式，4 个风扇	3368 (3483)	186	2088	169	1368 (1384)	174
自动固件模式，8 个风扇	3441 (3499)	366	N/A	N/A	1380 (1400)	342
自动固件模式，12 个风扇	3454 (3485)	546	N/A	N/A	1456 (1480)	510
自动固件模式，16 个风扇	3500 (3679)	726	N/A	N/A	1488 (1504)	678
自动硬件模式，4 个风扇	2567 (2581)	115	N/A	N/A	1120 (1132)	103
自动硬件模式，8 个风扇	2599 (2613)	227	N/A	N/A	1188 (1196)	203
自动硬件模式，12 个风扇	2652	339	N/A	N/A	1316	303

4 说明 8 位分辨率资源使用的表。如果被 10 位配置占用的资源有差异，那些资源将在括号内显示。

5 对于硬件模式，控制循环周期功能使用的资源不包括在内。

直流和交流的电气特性

除非另有说明，否则这些规范的适用条件是：-40°C ≤ T_A ≤ 85 °C 且 T_J ≤ 100 °C。电压范围为 1.71 V 到 5.5 V。

直流特性

PSoC 3/PSoC 5LP（PWM 时钟 — 6 MHz，BUS_CLK — 24 MHz）

参数	说明	最小值	典型值[6]	最大值	单位
Idd	组件电流消耗（手动模式，4个风扇）				
	8位分辨率	–	108	–	µA
	10位分辨率	–	139	–	µA
	组件电流消耗（手动模式，8个风扇）				
	8位分辨率	–	194	–	µA
	10位分辨率	–	260	–	µA
	组件电流消耗（手动模式，12个风扇）				
	8位分辨率	–	307	–	µA
	10位分辨率	–	395	–	µA
	组件电流消耗（手动模式，16个风扇）				
	8位分辨率	–	392	–	µA
	10位分辨率	–	505	–	µA
	组件电流消耗（自动固件模式，4个风扇）				
	8位分辨率	–	180	–	µA
	10位分辨率	–	221	–	µA
	组件电流消耗（自动固件模式，8个风扇）				
	8位分辨率	–	264	–	µA
	10位分辨率	–	319	–	µA
	组件电流消耗（自动固件模式，12个风扇）				
	8位分辨率	–	329	–	µA

6 不包含设备 I/O 和时钟分配的电流。布线条件、转速计输入的频率、温度等其他因素对电流消耗也有影响。这些值是在温度为 25 °C 条件下的值。

参数	说明	最小值	典型值[6]	最大值	单位
	10位分辨率	–	428	–	µA
Idd（续）	组件电流消耗（自动固件模式，16个风扇）				
	8位分辨率	–	385	–	µA
	10位分辨率	–	508	–	µA
	组件电流消耗（自动硬件模式，4个风扇）				
	8位分辨率	–	225	–	µA
	10位分辨率	–	269	–	µA
	组件电流消耗（自动硬件模式，8个风扇）				
	8位分辨率	–	417	–	µA
	10位分辨率	–	518	–	µA
	组件电流消耗（自动硬件模式，12个风扇）				
	8位分辨率	–	636	–	µA

PSoC 3/PSoC 5LP（PWM 时钟 — 12 MHz，BUS_CLK — 24 MHz）

参数	说明	最小值	典型值[7]	最大值	单位
Idd	组件电流消耗（手动模式，4个风扇）				
	8位分辨率	–	169	–	µA
	10位分辨率	–	233	–	µA
	组件电流消耗（手动模式，8个风扇）				
	8位分辨率	–	310	–	µA
	10位分辨率	–	439	–	µA
	组件电流消耗（手动模式，12个风扇）				
	8位分辨率	–	498	–	µA
	10位分辨率	–	677	–	µA
	组件电流消耗（手动模式，16个风扇）				
	8位分辨率	–	641	–	µA

7 不包含设备 I/O 和时钟分配的电流。布线条件、转速计输入的频率和温度等其他因素也影响到电流消耗。这些值是在温度为 25 °C 时的值。

参数	说明	最小值	典型值[7]	最大值	单位
	10位分辨率	–	871	–	µA
Idd（续）	组件电流消耗（自动固件模式，4个风扇）				
	8位分辨率	–	246	–	µA
	10位分辨率	–	307	–	µA
	组件电流消耗（自动固件模式，8个风扇）				
	8位分辨率	–	360	–	µA
	10位分辨率	–	486	–	µA
	组件电流消耗（自动固件模式，12个风扇）				
	8位分辨率	–	512	–	µA
	10位分辨率	–	695	–	µA
	组件电流消耗（自动固件模式，16个风扇）				
	8位分辨率	–	618	–	µA
	10位分辨率	–	870	–	µA
	组件电流消耗（自动硬件模式，4个风扇）				
	8位分辨率	–	403	–	µA
	10位分辨率	–	492	–	µA
	组件电流消耗（自动硬件模式，8个风扇）				
	8位分辨率	–	756	–	µA
	10位分辨率	–	951	–	µA
	组件电流消耗（自动硬件模式，12个风扇）				
	8位分辨率	–	1162	–	µA

PSoC 3/PSoC 5LP（PWM 时钟 — 24 MHz，BUS_CLK — 24 MHz）

参数	说明	最小值	典型值[8]	最大值	单位
Idd	组件电流消耗（手动模式，4个风扇）				
	8位分辨率	–	290	–	µA

8 不包含设备 I/O 和时钟分配的电流。布线条件、转速计输入的频率、温度等其他因素也影响到电流消耗。这些值是在温度为 25 °C 时的值。

参数	说明	最小值	典型值[8]	最大值	单位
	10位分辨率	–	417	–	µA
Idd（续）	组件电流消耗（手动模式，8个风扇）				
	8位分辨率	–	545	–	µA
	10位分辨率	–	798	–	µA
	组件电流消耗（手动模式，12个风扇）				
	8位分辨率	–	887	–	µA
	10位分辨率	–	1243	–	µA
	组件电流消耗（手动模式，16个风扇）				
	8位分辨率	–	1150	–	µA
	10位分辨率	–	1607	–	µA
	组件电流消耗（自动固件模式，4个风扇）				
	8位分辨率	–	354	–	µA
	10位分辨率	–	494	–	µA
	组件电流消耗（自动固件模式，8个风扇）				
	8位分辨率	–	614	–	µA
	10位分辨率	–	906	–	µA
	组件电流消耗（自动固件模式，12个风扇）				
	8位分辨率	–	897	–	µA
	10位分辨率	–	1246	–	µA
	组件电流消耗（自动固件模式，16个风扇）				
	8位分辨率	–	1089	–	µA
	10位分辨率	–	1671	–	µA
	组件电流消耗（自动硬件模式，4个风扇）				
	8位分辨率	–	753	–	µA
	10位分辨率	–	938	–	µA
	组件电流消耗（自动硬件模式，8个风扇）				
	8位分辨率	–	1446	–	µA
	10位分辨率	–	1826	–	µA
	组件电流消耗（自动硬件模式，12个风扇）				

参数	说明	最小值	典型值[8]	最大值	单位
	8位分辨率	–	2220	–	µA

PSoC 4（PWM 时钟 — 6 MHz，BUS_CLK — 24 MHz）

参数	说明	最小值	典型值	最大值	单位
Idd	组件电流消耗（手动模式，4个风扇）	–	605	–	µA

PSoC 4（PWM 时钟 — 12 MHz，BUS_CLK — 24 MHz）

参数	说明	最小值	典型值	最大值	单位
Idd	组件电流消耗	–	900	–	µA

PSoC 4（PWM 时钟 — 6 MHz，BUS_CLK — 48 MHz）

参数	说明	最小值	典型值	最大值	单位
Idd	组件电流消耗	–	670	—	µA

PSoC 4（PWM 时钟 — 12 MHz，BUS_CLK — 48 MHz）

参数	说明	最小值	典型值	最大值	单位
Idd	组件电流消耗	–	1010	–	µA

PSoC 4（PWM 时钟 — 24 MHz，BUS_CLK — 48 MHz）

参数	说明	最小值	典型值	最大值	单位
Idd	组件电流消耗	–	1200	–	µA

交流规范

PSoC 3/PSoC 5LP

参数	说明	最小值	典型值	最大值	单位
f_{pwm_clk}	输入时钟频率[9]				
	手动模式，8位分辨率，4个风扇	–	–	51	MHz
	手动模式，8位分辨率，8个风扇	–	–	45	MHz
	手动模式，8位分辨率，12个风扇	–	–	51	MHz
	手动模式，8位分辨率，16个风扇	–	–	47	MHz
	手动模式，10位分辨率，4个风扇	–	–	44	MHz
	手动模式，10位分辨率，8个风扇	–	–	43	MHz
	手动模式，10位分辨率，12个风扇	–	–	43	MHz
	手动模式，10位分辨率，16个风扇	–	–	43	MHz
	自动固件模式，8位分辨率，4个风扇	–	–	51	MHz
	自动固件模式，8位分辨率，8个风扇	–	–	46	MHz
	自动固件模式，8位分辨率，12个风扇	–	–	46	MHz
	自动固件模式，8位分辨率，16个风扇	–	–	56	MHz
	自动固件模式，10位分辨率，4个风扇	–	–	44	MHz
	自动固件模式，10位分辨率，8个风扇	–	–	44	MHz
	自动固件模式，10位分辨率，12个风扇	–	–	44	MHz
	自动固件模式，10位分辨率，16个风扇	–	–	44	MHz
	自动硬件模式，8位分辨率，4个风扇	–	–	55	MHz
	自动硬件模式，8位分辨率，8个风扇	–	–	44	MHz
	自动硬件模式，8位分辨率，12个风扇	–	–	46	MHz
	自动硬件模式，10位分辨率，4个风扇	–	–	50	MHz
	自动硬件模式，10位分辨率，8个风扇	–	–	50	MHz
f_{tach_clk}	转速计时钟频率	–	500	–	kHz
转速计分辨率	转速计计数器分辨率	–	16	–	位

9 这些值提供了各个风扇的最大安全工作脉冲宽度调制器（PWM）频率。可以在更高的时钟频率运行组件，在该频率将需要使用静态时序分析结果验证时序要求。

参数	说明	最小值	典型值	最大值	单位
f_{tach}	转速计速度	500[10]	–	25000	RPM

PSoC 4

参数	说明	最小值	典型值	最大值	单位
f_{pwm_clk}	输入时钟频率[11]	–	–	48	MHz

组件勘误表

本节列出了组件的已知问题。

赛普拉斯ID	组件版本	问题	解决方案
191257	3.0	该组件的版本已被修改，但没有更改PSoC Creator 3.0 SP1中的版本编号。更多有关信息，请参见基础知识文章KBA94159（网页地址：www.cypress.com/go/kba94159）。	没有解决方案。您必须将该组件更新为最新版本。

10 低于最小速度的速度意味着停止。

11 这些值提供了各个风扇的最大安全工作脉冲宽度调制器（PWM）频率。可以在更高的时钟频率运行组件，在该频率将需要使用静态时序分析结果验证时序要求。

组件更新

本节列出了该组件各版本中的主要更改内容。

版本	更改说明	更改原因/影响
3.10	纠正了PSoC Creator 3.0 SP1中组件更改内容。	纠正了组件勘误表 — 赛普拉斯ID 191257。
3.0.b	组件版本3.0已经过期，不再使用。	该组件的版本已被修改，但没有更改PSoC Creator 3.0 SP1中的版本编号。
3.0.a	更新了警报输出、风扇停顿/电机锁定以及速度调节部分中数据手册说明内容。	添加了关于警报如何执行的详细信息。
3.0	更新了资源、API用途、功能说明、 直流和交流电气特性的部分。	
	解决了FanController_GetActualSpeed() API。	该函数有时会返回错误的测速值。
	删除了有关在PSoC 3的EOC脉冲后特定时间内调用FanController_GetActualSpeed()的注释。	该注意项的原因与CPU和DMA同时访问多字节数据的操作相关。该组件的当前版本解决了这个问题。
	添加了控制循环周期的说明内容。	
	添加了两个新的API函数：FanController_SetSaturation()和FanController_PID()。	需要这两个函数支持新组件的模式。
	将函数名称“FanController_OverrideHardwareControl()”修改为“FanController_OverrideAutomaticControl()”。	这是由于旧名称不符合新模式的介绍内容。
	清除了PSoC 5支持。	
	添加了PSoC 4系列器件的支持。	提供了PSoC 4的风扇控制功能。
	将固件（CPU）模式重新命名为手动模式。	该模式不会执行实际的风扇控制，但该算法由用户实现。新模式 — 自动固件（CPU）使用内部PID控制算法来控制各个风扇。
	新的控制模式 — 已将自动固件（CPU）添加到该组件内。该模式的设置取决于PID控制算法。	新性能。
2.30.a	将标签“阻尼系数（sec）”重新命名为“控制循环周期（sec）”。	由于相应的参数是时间数量，所以术语“控制循环周期”是更好的。
2.30	添加了“MISRA合规性”章节。	该组件未通过MISRA合规性验证。
	已将风扇控制器的时钟和DMA组件更新为最新版本。	

版本	更改说明	更改原因/影响
2.20	更新了FanController_GetFanSpeedStatus()、FanController_GetFanStallStatus()和FanController_GetAlertSource()函数中说明的“其他影响”一节。	
	解决了关于警报引脚的问题，其与数据手册中所述的不同。	在每个“eoc”的输入时钟周期内，警报信号被置为高电平；它预计将一直被置为高电平，直到被FanController_GetFanSpeedStatus()或FanController_GetFanStallStatus()函数清除。现在，一直将它置为高电平，直到它被FanController_GetAlertSource()清除。
	解决了关于固件模式中组件运行的错误，其中组件的配置支持6极电机。	由于实现组件状态机的过程中发生了错误，因此该组件不在这种配置中运行。
	根据定制器验证未使用（隐藏）组件设置进行修复错误。	现在，只在风扇表中可见的行值才可检查其正确性。
	根据定制器生成的初始占空比的不正确值进行修复错误。	
2.10	更新了组件特性数据。	
	添加了PSoC 5LP支持	
2.0	添加了组件特性数据。	
	更改了阻尼系数的功能。	在之前的版本中，阻尼系数用于指定每个风扇各速度测量之间的延迟，且未使用任何单位指定延迟值。在该版本中，阻尼系数用于指定所有风扇的速度测量的开始时间之间的延迟。此外，该延迟现在使用秒为单位进行指定。
	在配置中，向每个风扇添加了新参数Initial RPM（初始 RPM）。	该参数指定在组件启动时特定风扇的旋转的近似速度。在之前的版本中，所有风扇都使用最大速度进行初始化。
	添加了用于选择该组件支持的电机类型的新功能。	该参数指定每次风扇旋转一周的高低脉冲数量。4极电机为两个高低脉冲，6极电机为三个高低脉冲。
	已使用添加外部时钟源以驱动内部脉冲宽度调制器的功能扩展了组件符号。可在定制器的GUI中访问该选项。	新增组件性能。允许连接到一个外部时钟。基于配置中源时钟的值和脉冲宽度调制器的分辨率，可调节脉冲宽度调制器输出频率。
	添加了新性能，该功能可在定制器的GUI中进行选择，并且用于将组件输入和输出显示为总线。	现在一系列的tach1-tachN、fan1-fanN和bank1-bankN连接可以显示为总线。这实现了原理图上空间的节约，因为此选项减小了组件符号的大小。
	已向API中添加了Keil函数可重入支持。	添加该功能，以便使客户能够指定任意单个所生的函数为可重入。

版本	更改说明	更改原因/影响
	删除了过时函数名称 — FanController_OverrideClosedLoop()，其曾经是FanController_OverrideHardwareControl()的简单#define。	因该函数在之前的版本中已被标记为已过，所以本版本中删除了它。
1.20	1. 进行了更新，以兼容于PSoC Creator v2.0 2. 被重新分类为“概念”组件 3. 修改了SetDesiredSpeed() API，以提高准确性	
1.10	1. 修复了SetDesiredSpeed() API 2. 向转速计输入添加了短时脉冲滤波器 3. 风扇控制术语由开/闭环更改为固件/硬件控制方法 4. 更新了符号颜色和大小 5. 更新（减少）了资源利用率 6. 删除了电源管理API的参考（不支持）	
1.0	第一版	

©赛普拉斯半导体公司，2014。此处所包含的信息可能会随时更改，恕不另行通知。除赛普拉斯产品内嵌的电路以外，赛普拉斯半导体公司不对任何其他电路的使用承担任何责任。也不会根据专利权或其他权利以明示或暗示的方式授予任何许可。除非与赛普拉斯签订了明确的书面协议，否则赛普拉斯产品不保证产品能够用于或适用于医疗、生命支持、救生、关键控制或安全应用领域。此外，对于可能发生运转异常和故障并对用户造成严重伤害的生命支持系统，赛普拉斯不授权将其产品用作此类系统的关键组件。若将赛普拉斯产品用于生命支持系统，则表示制造商将承担因此类使用而招致的所有风险，并确保赛普拉斯免于因此而受到任何指控。

PSoC®是赛普拉斯半导体公司的注册商标，PSoC Creator™和 Programmable System-on-Chip™是赛普拉斯半导体公司的商标。该处引用的所有其它商标或注册商标归其各自所有者所有。

所有源代码（软件和/或固件）均归赛普拉斯半导体公司（赛普拉斯）所有，并受全球专利法规（美国和美国以外的专利法规）、美国版权法以及国际条约规定的保护和约束。赛普拉斯据此向获许可者授予适用于个人的、非独占性、不可转让的许可，用以复制、使用、修改、创建赛普拉斯源代码的派生作品、编译赛普拉斯源代码和派生作品，并且其目的只能是创建自定义软件和/或固件，以支持获许可者仅将其获得的产品依照适用协议规定的方式与赛普拉斯集成电路配合使用。除上述指定的用途之外，未经赛普拉斯的明确书面许可，不得对此类源代码进行任何复制、修改、转换、编译或演示。

免责声明：赛普拉斯不针对此材料提供任何类型的明示或暗示保证，包括（但不仅限于）针对特定用途的适销性和适用性的暗示保证。赛普拉斯保留在不做出通知的情况下对此处所述材料进行更改的权利。赛普拉斯不对此处所述之任何产品或电路的应用或使用承担任何责任。对于合理预计可能发生运转异常和故障，并对用户造成严重伤害的生命支持系统，赛普拉斯不授权将其产品用作此类系统的关键组件。若将赛普拉斯产品用于生命支持系统中，则表示制造商将承担因此类使用而招致的所有风险，并确保赛普拉斯免于因此而受到任何指控。

产品使用可能受适用于赛普拉斯软件许可协议的限制。