初版

热敏电阻数据手册 热敏电阻 V 2.00

该数据手册中包含了初步信息。

MUM		PSoC® 模块				API 存储器 （字节）		引脚
		数字	模拟 CT	模拟 SC	抽取滤波器	闪存	RAM	
受支持的器件包括：CY8C24x23A、 CY8C27x43、 CY8C24x94、 CY8C29x66、 CY8C28x23、 CY8C28x33、 CY8C28x43、 CY8C28x45、 CY8C28x52								
配置	路由参考电压，RefLo ≠ VSS	2	2	1	1	Code_Size* + LUT_Size**	18	3
	直接参考电压，RefLo = VSS	2	1	1	1	Code_Size* + LUT_Size*	18	3

注意：

* Code_Size:

基本器件	代码大小 （字节）
CY8C24x23A、 CY8C27x43	1056
CY8C24x94、 CY8C29x66	1101
CY8C28x23、 CY8C28x33、 CY8C28x43、 CY8C28x45、 CY8C28x52	1081

** LUT_Size：根据下面的公式来确定 LUT （查找表）大小 （单位为字节）：

LUT_Size = （（最高温度 – 最低温度）+ 1）* 4；

在 “ 热敏电阻向导 ” 一节中对最低温度和最高温度进行了定义。

特性与概述

通过测量电阻值，并使用一个带线性平滑曲线的表格计算温度，因此可使用热敏电阻用户模块进行测量某热敏电阻的温度。使用 Steinhart-Hart 公式创建该表格。以下是该用户模块的重要特性：

- 提供了一个端口，这样通过使用比率度量方法可以得出热敏电阻的温度
- 使用线性曲线可以计算温度
- 自动计算时钟频率
- 补偿偏移误差

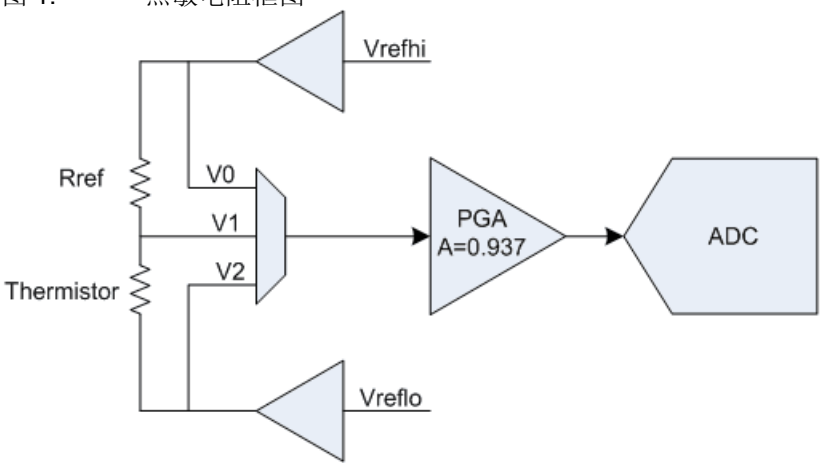
快速入门

1. 选择并放置用户模块目录内温度类别中的热敏电阻用户模块。
2. 在多用户模块 （MUM）向导中，选择路由参考或直接参考的配置。
3. 在 **Workspace Explorer** （工作区浏览器）中右键单击热敏电阻用户模块，以访问热敏电阻向导（本数据手册后面的内容将对向导进行介绍）。
4. 输入计算温度时使用的参考电阻参数。
5. 输入最小、中间和最大这三个点的温度 / 电阻比例的参数 （单位为摄氏度 / 欧姆）。
6. 输入将要连接输入热敏电阻的引脚。点击 **OK**。
7. 输入用户模块参数。
8. 生成应用，并切换到 **Application Editor** （应用编辑器）。
9. 根据要求调整示例代码。
10. 通过使用 PSoC Designer™ 生成的十六进制 hex 文件对目标电路板上的 PSoC 进行编程。

功能说明

热敏电阻的测量方法基于赛普拉斯应用笔记， AN2017 — PSoC 1 热敏电阻的温度测量。

图 1. 热敏电阻框图



在图 1 中，测量热敏电阻值的公式为：

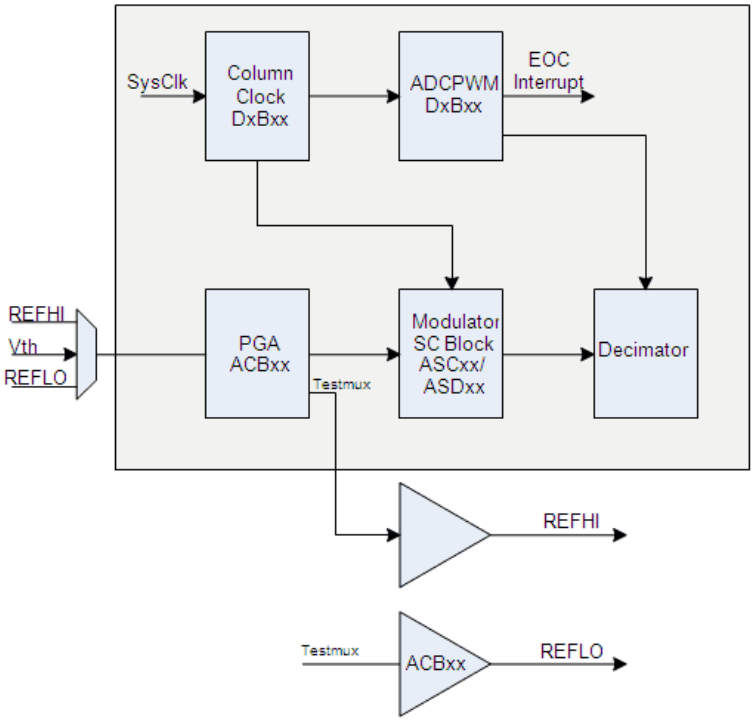
Rthermistor = Rref * （V1-V2）/ （V0-V1）

当使用测得的电压计算出热敏电阻值后，可以通过下面的二元组合方式计算温度：从温度 / 电阻表中查找结果的整数部分，然后使用线性平滑曲线进行模拟上述各点，得到十进制的分辨率。

热敏电阻是整数值 （无符号长），单位为 1 欧姆的 1/10。温度值是 1 摄氏度百分之一单位的整数值。

下图显示的是热敏电阻用户模块的 PSoC 1 硬件模块框图。

图 2. 硬件模块框图



热敏电阻用户模块的主要组成部件分别为：一个可编程增益放大器 （PGA）、一个增量模数转换器（ADC）、两个模拟缓冲器和一个复用器。

一个 CT 模块被配置为 PGA。将该 CT 模块的 testmux 配置为 REFHI，并将它与列模拟总线连接起来。将其他 CT 模块配置为 RefMux 以输出 REFLO。将 PGA 增益配置为 0.937。

一个单一的 SC 模块被配置为一阶 ADC 调制器。调制器的输出与抽取滤波器的输入端连接起来。

使用一个计数器生成 ADC 的列时钟。配置另一个计数器，用于生成 ADC 的门脉冲。该 ADC 被配置为 13 位的分辨率，并能够使用最大为 2.0 MHz 的列时钟频率。这样便要求一列时钟分频器的最小值为 12，并使最大采样 ADC 的采样率为 61 Hz。使用三个采样进行测量温度时，最大的更新速率为每秒 20.3 °C。

直流和交流电气特性

下列各值表示预计的性能，它们是根据初始特性数据得到的。

表 1. 热敏电阻的直流和交流电气特性

参数	说明	条件	最小值	典型值	最大值	单位
热敏电阻的特性						
R_{Error}	电阻测量误差	参考电阻值 = 10K，容差为 0.1% ；热敏电阻的阻值： 1K 10K 100K		 0.50 0.25 0.50	 – – –	%
I_{oper}	工作电流	低功耗 中等功耗 高功耗	–	272 1380 5452	–	µA
T_{Error}	温度误差	热敏电阻 = 10 k ；参考电阻 = 10k，容差 = 0.1% 温度为 -40°C 时的 R_{TH} 温度为 25°C 时的 R_{TH} 温度为 400°C 时的 R_{TH}	 –0.2 – 0.15 –0.2	 ±0.1 ±0.1 ±0.1	 0.2 0.15 0.2	°C

放置

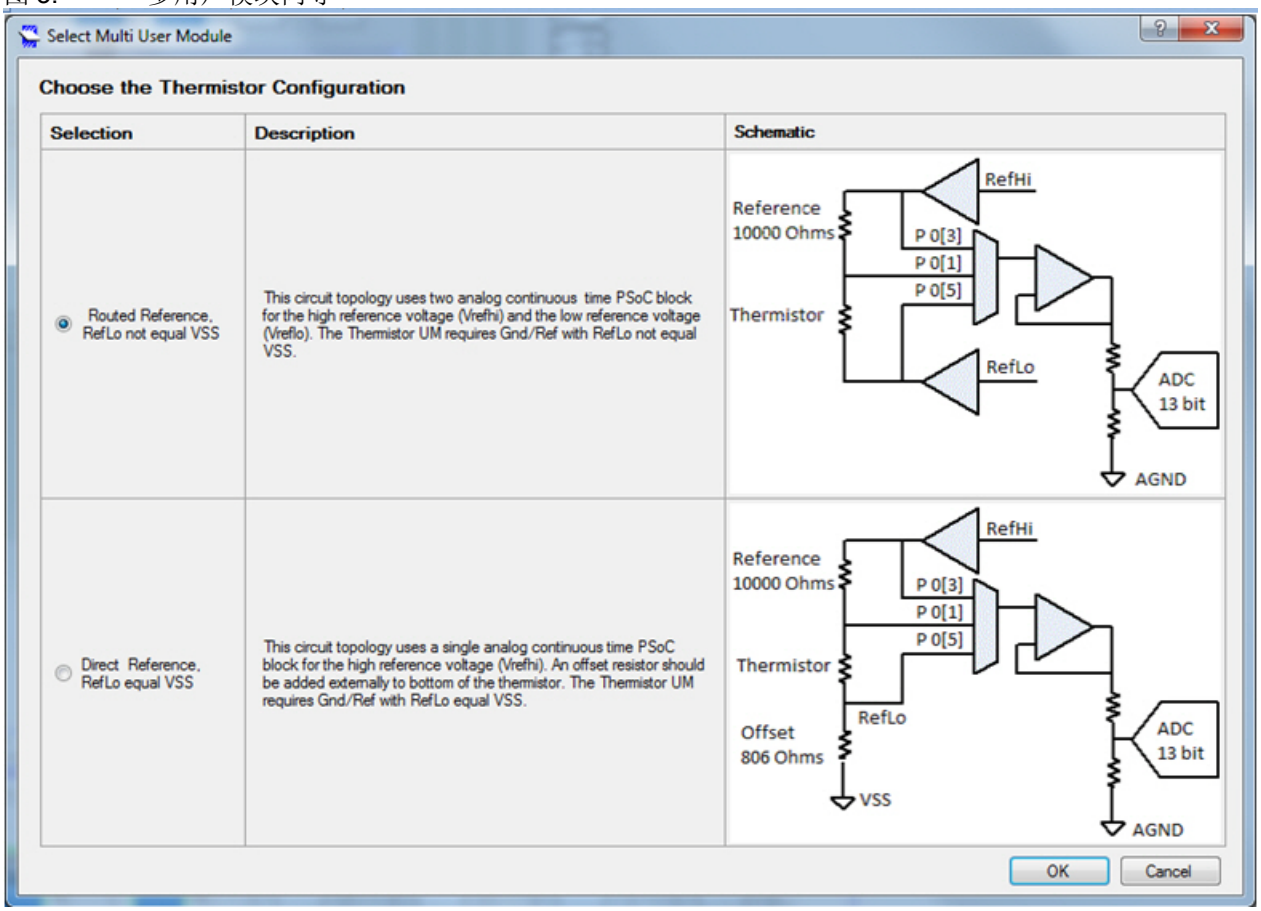
热敏电阻是一个多用户模块 （MUM）。 RefLo ≠ VSS 的热敏电阻需要两个 CT 模块、一个 SC 模块和两个数字模块。 RefLo = VSS 的热敏电阻需要一个 CT 模块、一个 SC 模块和两个数字模块。

在一个项目中只能放置一个用户模块实例。

多用户模块向导

选择并放置用户模块目录内温度类别中的热敏电阻用户模块。您可以通过 MUM 向导 （请看见下面的截图）选择配置。

图 3. 多用户模块向导



MUM 具有下面两种配置方式：

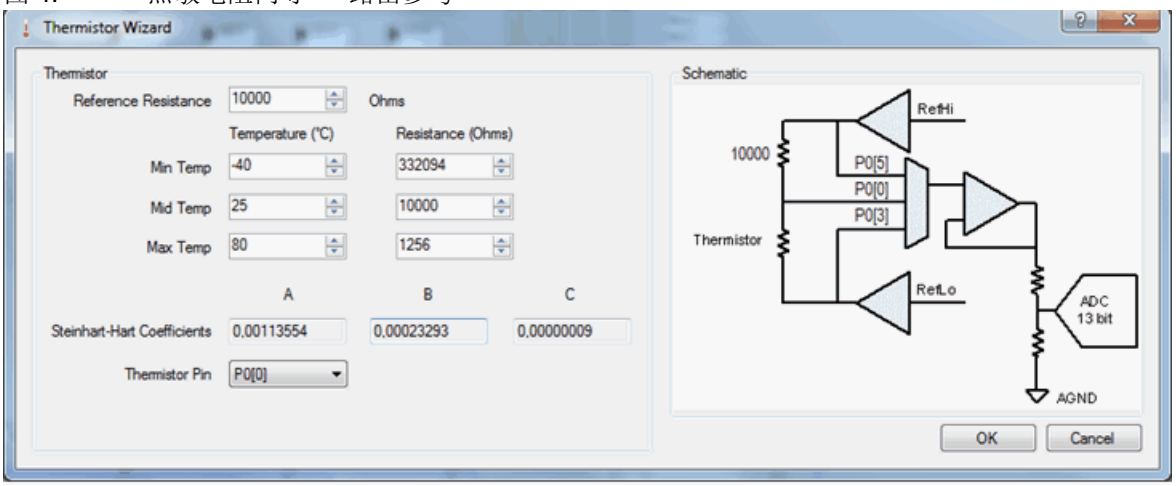
1. 路由参考， RefLo ≠ VSS：该电路拓扑结构分别为高参考电压 （Vrefhi）和低参考电压 （Vreflo）使用了两个模拟连续时序 PSoC 模块。热敏电阻用户模块要求 Gnd/Ref 中 RefLo ≠ VSS。
2. 直接参考， RefLo = VSS：该电路拓扑结构为高参考电压 （Vrefhi）使用了一个模拟连续时序 PSoC 模块。将一个外部偏移电阻添加到热敏电阻的底部。热敏电阻用户模块要求 Gnd/Ref 中 RefLo = VSS。

请单击 **OK** 进行放置热敏电阻用户模块。

热敏电阻向导

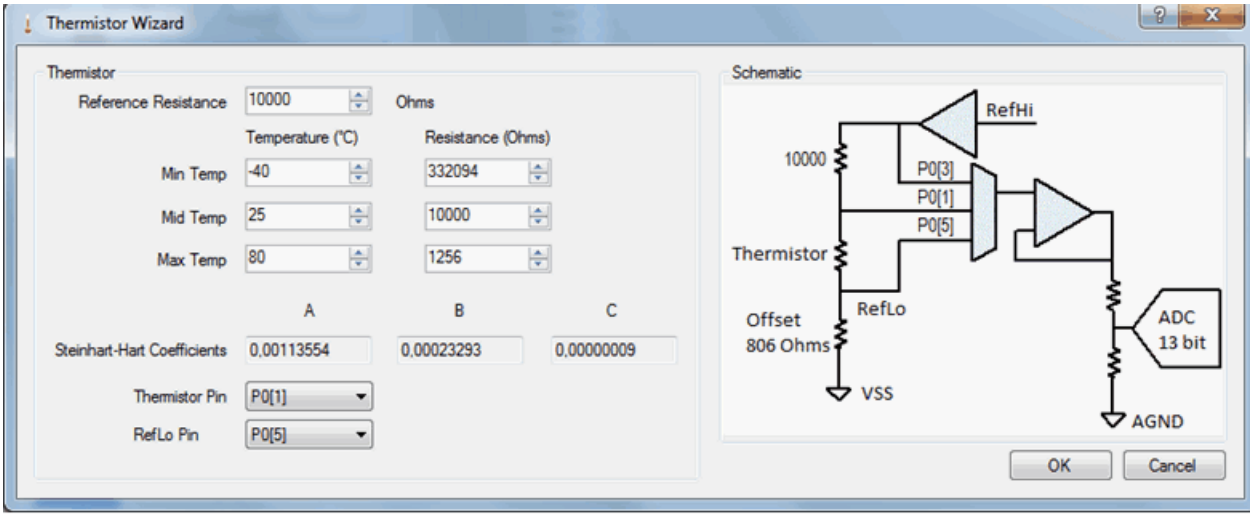
在 **Chip Editor** 的 **Workspace Explorer** 中右键单击用户模块菜单，然后选择 “Thermistor Wizard” （热敏电阻向导），即可打开热敏电阻向导。选择 MUM 作为路由参考使用时，（RefLo 不等于 VSS），向导如下所示。

图 4. 热敏电阻向导 — 路由参考



当选择 MUM 作为外部连接的参考 （RefLo = VSS）时，该向导的截图如下所示。在框图图像和可用引脚选择上存在差异。

图 5. 热敏电阻向导 — 外部链接参考



向导会生成一个 1.0 度间隔的检查表。检查表中点的数量取决于 “Min Temp” （最低温度）和 “Max Temp” （最高温度）参数，但不能超过 141 个。 MUM 向导具有下面各参数：

Reference Resistance （参考电阻）

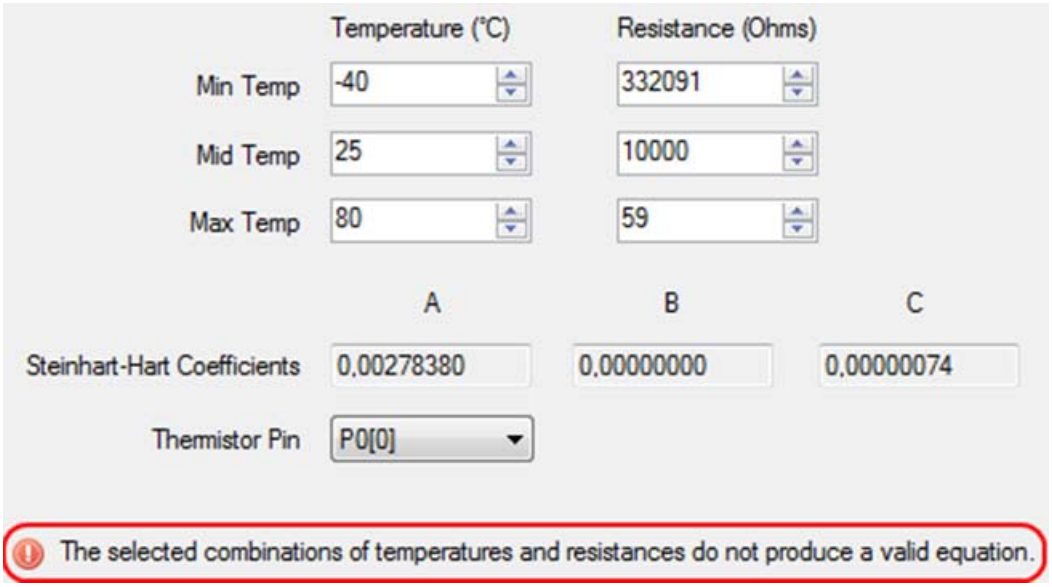
根据参考电阻参数进行设置计算温度时所使用的参考电阻值。阻值范围为 1000 ~ 30000。默认值为 10000。

Min Temp （最低温度）、 Mid Temp （中间温度）和 Max Temp （最大温度）

最低温度、中间温度和最高温度等参数分别用于设置三个温度和电阻点的最低、中间和最高值。温度范围为 -40 °C ~ 100 °C。默认情况下，最低值为 -40 °C，中间值为 25 °C，最大值为 80 °C。电阻值

范围为 50 到 5.0E6 Ω。默认情况下，温度最低时电阻为 332094 Ω，中间温度时电阻为 10000 Ω，另外温度最高时电阻为 1256 Ω。

并非所有温度与电阻的组合都能生成有效的 Steinhart-Hart 公式。如果输入的值生成的公式无效，将在向导中生成以下错误：



当使用热敏电阻数据手册中介绍的参考值或使用预先测量的准确值时，不会出现这种情况。

Steinhart-Hart Coefficients （Steinhart-Hart 系数）

Steinhart-Hart 系数是根据最低温度、中间温度、最高温所输入的三组温度和电阻值计算得到的。电阻值和 Steinhart-Hart 系数是相互依存的。如果您更改了任何电阻值， Steinhart-Hart 系数也被更新。同样，如果您更改了 Steinhart-Hart 值，电阻值也会被更新。

Vref Pins （Vref 引脚）

对于内部路由参考，您可以选择 Vref 引脚 {P0[3] 和 P0[5]} 或者引脚 {P0[2] 和 P0[4]}。根据用户模块的放置情况，该选择可能发生变化。 UM 向导中提供了该信息，从而能够得到将 Vref 引脚和热敏电阻引脚结合起来的方法。

Thermistor Pins （热敏电阻引脚）

热敏电阻引脚设置的引脚将热敏电阻结和参考电阻器连接起来。

RefLo Pin （RefLo 引脚）

只有将 MUM 选为外部连接参考 （RefLo = VSS）时，才能对 RefLo 引脚进行布线。

原理图

如果可用，可以使用 “ 参考电阻 ”、“Vref 引脚 ”、“ 热敏电阻引脚 ” 和 “RefLo 引脚 ” 值进行更新原理图。

向导按键

用户模块向导显示了四个按键：

- OK — 该按键用于检查向导的参数设置。如果正确定义了各参数，则向导将存储各参数并关闭；否则，向导会显示相应的警告信息，并且不会关闭。
- “Cancel” — 该按键用于关闭向导，并不会保存各参数。
- “Close” — 该按键可关闭向导，并不会保存各参数。

■ “Help” — 使用该按键可调用热敏电阻用户模块向导的帮助信息。它描述了用户模块向导的特性。

参数和资源

用户模块被放置后，必须配置下面各参数以进行正常的操作：热敏电阻引脚、 RefLo 引脚、温度采样率。

Thermistor Pins （热敏电阻引脚）

该参数用于选择热敏电阻所连接的引脚。可按照 MUM 向导中的 “Vref 引脚 ” 选项选择引脚。引脚范围为 P0[0] 到 P0[7]，默认情况下被设置为 P0[0]。

RefLo 引脚

该参数用于选择 RefLo 所连接的引脚。只有选中了外部路由 RefLo 的配置时，才能选择该引脚。可按照向导中介绍的 “Vref 引脚 ” 选项选择引脚。“Vref 引脚 ” 参数所选的各端口引脚都不会列在该参数中。引脚范围为 P0[0] 到 P0[7]，默认情况下被设置为 P0[5]。

温度采样率

该参数用于设置 ADC 的采样率。取值范围从 20.3 （整数值 = 20 sps）到 1.0 sps。该采样率是 ADC 采样率 （60.1 sps）的 1/3，这是因为计算温度时使用了三个测量结果。要保证 ADC 的最大列时钟频率 （为 2.0 MHz）符合规范要求。当采样率是本地交流线路频率的因数时，可获得最佳性能。

取值范围为 1 到 20。默认采样率同 13 位转换器的三个样本相对应，该转换器具有在 PWM 中进行 12 分频得到的列时钟 （其最大频率为 2.0 MHz）。

应用编程接口 （API）

所提供的应用编程接口 （API）子程序作为用户模块的一部分，设计人员通过它可以采用更高级的方式处理模块。本节指定了每个函数的接口，以及 “ 包含 ” 文件所提供的相关常量。

注意：

在这里，同所有用户模块 API 中的一样， A 和 X 寄存器的值可以通过调用 API 函数进行更改。如果在调用后需要 A 和 X 的值，则调用函数要保留在调用前的 A 和 X 的值。选择该 “ 寄存器易失 ” 策略是为了提高效率，自 PSoC Designer 1.0 版起已强制使用该策略。 C 编译器自动遵守该要求。汇编语言程序员也必须确保其代码遵循该策略。虽然一些用户模块 API 函数可以保持 A 和 X 不变，但是无法保证它们将来也会如此。

Thermistor_Start

说明：

启动热敏电阻用户模块的资源。根据热敏电阻的采样率参数，在定制器中计算功耗设置并将其保存为默认值。

热敏电阻的采样率	符号名称	默认值
≤ 5 sps	Thermistor_LOWPOWER	1
> 5 sps 且 ≤ 10 sps	Thermistor_MEDPOWER	2
> 10 sps	Thermistor_HIGHPOWER	3

C 语言原型：

```
void Thermistor_Start (void)
```

汇编：

```
lcall  Thermistor_Start
```

参数：

在定制器中计算功耗设置，并将其保存为默认值。

符号名称	数值
Thermistor_LOWPOWER （采样率 ≤ 5 sps）	1
Thermistor_MEDPOWER （采样率 > 5 sps 且 ≤ 10 sps）	2
Thermistor_HIGHPOWER （采样率 > 10 sps）	3

返回值：

无

其他影响：

无。

Thermistor_SetPower

说明：

将用户模块的功耗设置修改为 Thermistor_Start API 中设定的默认值。

C 语言原型：

```
void  Thermistor_SetPower(BYTE bPowerSetting)
```

汇编：

```
mov A, bPowerSetting
lcall  Thermistor_SetPower
```

参数：

bPowerSetting

符号名称	数值
Thermistor_LOWPOWER	1
Thermistor_MEDPOWER	2
Thermistor_HIGHPOWER	3

返回值：

无

其他影响：

无。

Thermistor_Stop

说明：

停止热敏电阻用户模块的资源。

C 语言原型：

```
void  Thermistor_Stop (void)
```

汇编：

```
lcall  Thermistor_Stop
```

参数：

无

返回值：

无

其他影响：

无。

Thermistor_StartConversion

说明：

该函数用于初始化测量热敏电阻温度的操作。启动该操作后，将在 ISR 中执行下列功能：复用并测量 Vrefhi/ Vth/ Vreflo 电压。为执行此序列的操作， ISR 将被作为一个状态机使用。

C 语言原型：

```
void Thermistor_StartConversion(void)
```

汇编：

```
lcall Thermistor_StartConversion
```

参数：

无。

返回值：

无

其他影响：

无。

Thermistor_fIsDataAvailable

说明：

该函数用于检查是否完成热敏电阻的转换。

C 语言原型：

```
BYTE  Thermistor_fIsDataAvailable(void)
```

汇编：

```
lcall  Thermistor_fIsDataAvailable
```

参数：

无

返回值：

如果转换未完整，将返回 0 ；如果转换已完成，则返回 1。

其他影响：

无。

Thermistor_iGetTemperature

说明：

返回最后一次转换的温度值。

C 语言原型：

```
INT Thermistor_iGetTemperature(void)
```

汇编：

```
lcall Thermistor_iGetTemperature
```

参数：

无

返回值：

0.01 °C 的 iTemperature 值。

其他影响：

无。

Thermistor_GetResistance

说明：

读取热敏电阻的电阻值，精确度为 0.1 Ω。使用该参数可校准操作。它是供内部使用的，通常不要求传送温度值。

C 语言原型：

```
void Thermistor_GetResistance(DWORD * pdwResistance)
```

汇编：

```
mov   A, >pdwResistance
mov   X, pdwResistance
lcall Thermistor_GetResistance
```

参数：

无

返回值：

pdwResistance：指向保留热敏电阻值的缓冲区的指针。 X 寄存器中装载返回缓冲器的地址。

其他影响：

无。

示例固件源代码

下面 C 语言的示例代码描述了热敏电阻用户模块的功能。

```
//-------------------------------------------------------------------------
// C main line
//-------------------------------------------------------------------------

#include <m8c.h>        // part specific constants and macros
#include "PSoCAPI.h"    // PSoC API definitions for all user modules

INT Temperature;
DWORD Resistance;

void main(void)
{
    M8C_EnableGInt;                          // Enable Global Interruptss
    Thermistor_Start();                      // Apply power to Thermistor
    Thermistor_StartConversion();           // Have Thermistor run

    for(;;)
    {
        while (!(Thermistor_fIsDataAvailable())); /* Loop until value ready */
        Temperature = Thermistor_iGetTemperature();/* get temperature */
        Thermistor_GetResistance(&Resistance); /* get resistance */

        /* Add user code here to use or display result */
    }
}
```

下面是汇编语言的示例代码：

```
;-------------------------------------------------------------------------
; Assembly main line
;-------------------------------------------------------------------------

include "m8c.inc"       ; part specific constants and macros
include "memory.inc"    ; Constants & macros for SMM/LMM and Compiler
include "PSoCAPI.inc"   ; PSoC API definitions for all User Modules

export _main

area bss (RAM,REL)
    Resistance:     BLK    4
    Temperature:    BLK    2

area text (ROM, REL)

_main:

    M8C_EnableGInt                  ; Enable Global Interrupts
    lcall Thermistor_Start          ; Apply power to Thermistor
    lcall Thermistor_StartConversion  ; Have Thermistor run continuously

terminate:
wait:
    lcall Thermistor_fIsDataAvailable ; Loop until value ready
```

```
    jz wait

    lcall Thermistor_iGetTemperature  ;  Get temperature
    mov [Temperature], X
    mov [Temperature + 1], A

    mov   A, >Resistance
    mov   X, Resistance
    lcall Thermistor_GetResistance    ;  Get resistance

    jmp terminate
```

向下兼容

在使用 PD5.4 打开过程中，热敏电阻 UM 版本 1.00 将保持不变 （不更新任何信息）。

根据常规的 UM 更新机制，热敏电阻 UM 版本 2.00 不会向后兼容。

版本历史记录

版本	创作者	说明
1.00	DHA	初始版本。
2.00	HPHA	1. 进行了重要的修改，以简化用户模块的操作和使用。 2. 取消了 ADC 特性。 3. 添加了对 CY8C24x23A 系列的支持。

注意： 在所有用户模块数据手册中， PSoC Designer 版本 5.1 都介绍了 “ 版本历史记录 ”。本数据手册介绍了当前和先前用户模块版本之间的区别的概述。