Application Note

TDA4x 及 AM6x 的热管理

摘要

Jacinto™ 7 TDA4 系列及 Sitara™ AM6x 处理器是 TI 基于 Keystone 架构推出的最新一代处理器。它们基于异构多核架构，集成了不同的应用内核，将不同的任务分配给相应的内核以进行并行处理，从而充分发挥 TI 处理器的优势。处理器中集成的计算能力越高，产生的发热就越高，如果超过热设计点并继续运行，可能会造成不可逆转的损坏。因此，为了在不影响处理器使用寿命的情况下充分利用计算能力，热管理变得非常重要。所有 TI 处理器都集成了片上温度传感器及用于热管理的专用模块 VTM（电压热管理）。片上温度传感器的基本工作原理是利用半导体材料的物理特性。当温度发生变化时，半导体材料的电阻或电势会发生变化，从而导致输出电信号发生变化。该变化由微控制器或其他电路读取和处理，从而实现温度测量。

VTM 模块还可以处理电压管理功能。本应用手册介绍了 TI 处理器中集成的 VTM 模块的功能，包括其工作原理、使用方法以及实现的软件和硬件保护方案。在讨论软件代码和特定应用示例时，使用 AM62A 处理器作为案例研究。可以将相应的代码更改和命令应用于全部 TDA4x 和 AM6x 系列处理器。但是，由于 SDK 版本更新，可能需要进行相应的编码路径修改。

内容

商标

所有商标均为其各自所有者的财产。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

1 VTM 模块

VTM（电压热管理）模块提供与集成温度传感器及用户编程热事件相关的控制、状态、中断和事件生成功能。在芯片热管理方面，我们主要关注其温度测量、警告发射和中断功能。图 1-1 显示了处理器内部的 VTM 方框图。VTM 与处理器的复位控制器直接连接，因为它可以直接发送命令来复位处理器。此外，VTM 还包括一组存储器映射寄存器，用于存储器件测试期间设置的器件特定工作电压 (AVSVNOM)。系统 AVS 软件可以使用这些值来调节器件的工作电压，从而实现更优工作条件（功耗和性能平衡）。

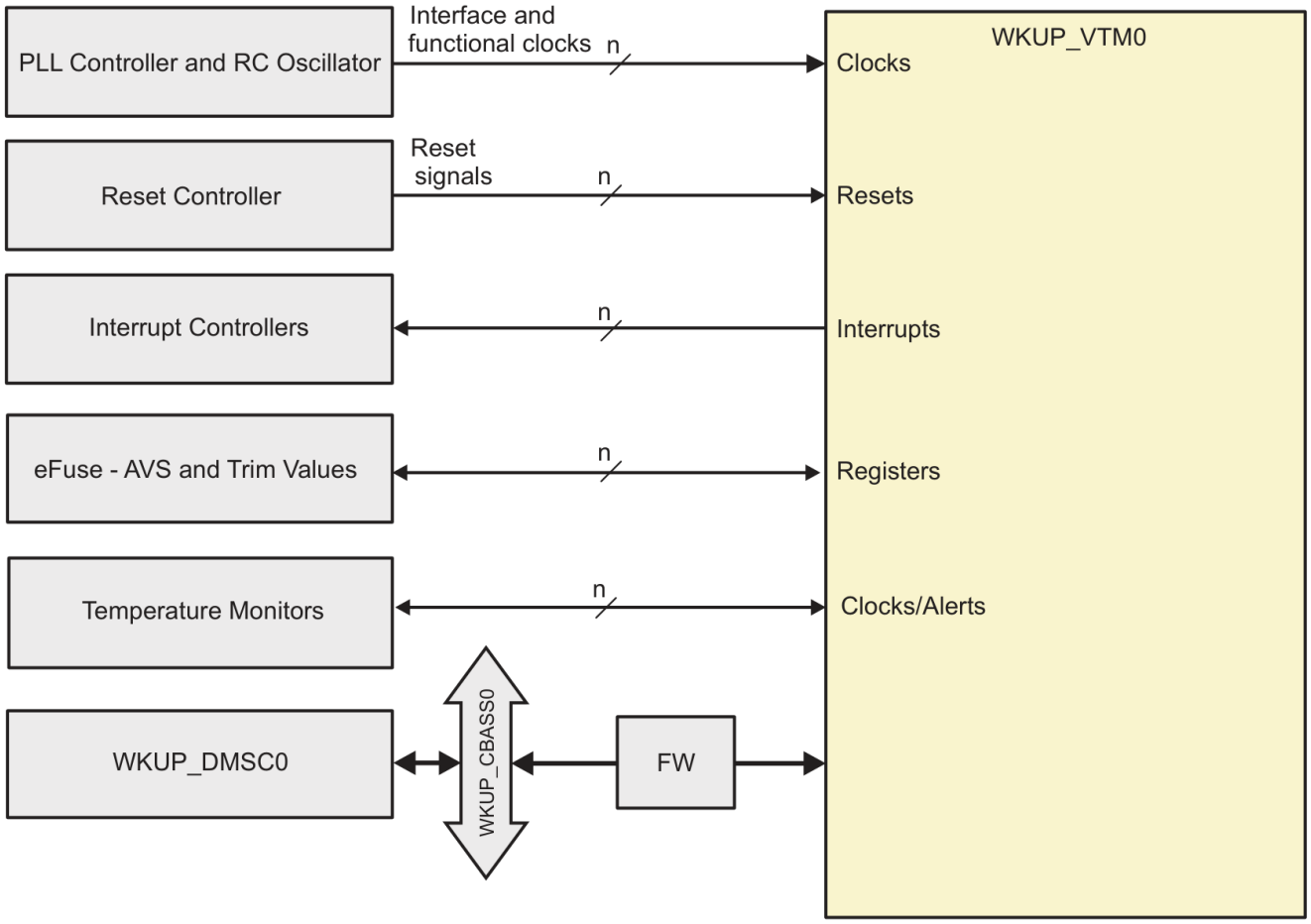

图 1-1. VTM 系统图

TI 有大量的 SOC 系列产品，VTM 系统连接非常相似。TI 有大量的 SOC 系列产品，VTM 系统连接非常相似。VTM 通常位于唤醒域 (WKUP) 中，这是 SOC 上电时首先开始工作的域。温度管理会在上电后立即干预，以确保整个 SOC 正常运行及其使用寿命。

1.1 VTM 模块说明

TI SOC 上 VTM 的典型布局如图 1-2 所示。这表明温度监测器（例如片上温度传感器）置于发热区域附近。VTM 模块经由内部连接控制芯片内的温度监测器。单个 VTM 可以通过寄存器控制多达八个温度传感器。由于温度传感器不会自行定期更新，因此 VTM 会定期启用温度传感器持续更新报告的温度数据。温度传感器返回的温度值由 VTM 寄存器捕获，并存储在 VTM 内的相应寄存器中。未启用传感器时，VTM 会将这些传感器保持在复位状态以省电并减少传感器使用量，从而更大限度地延长传感器使用寿命。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

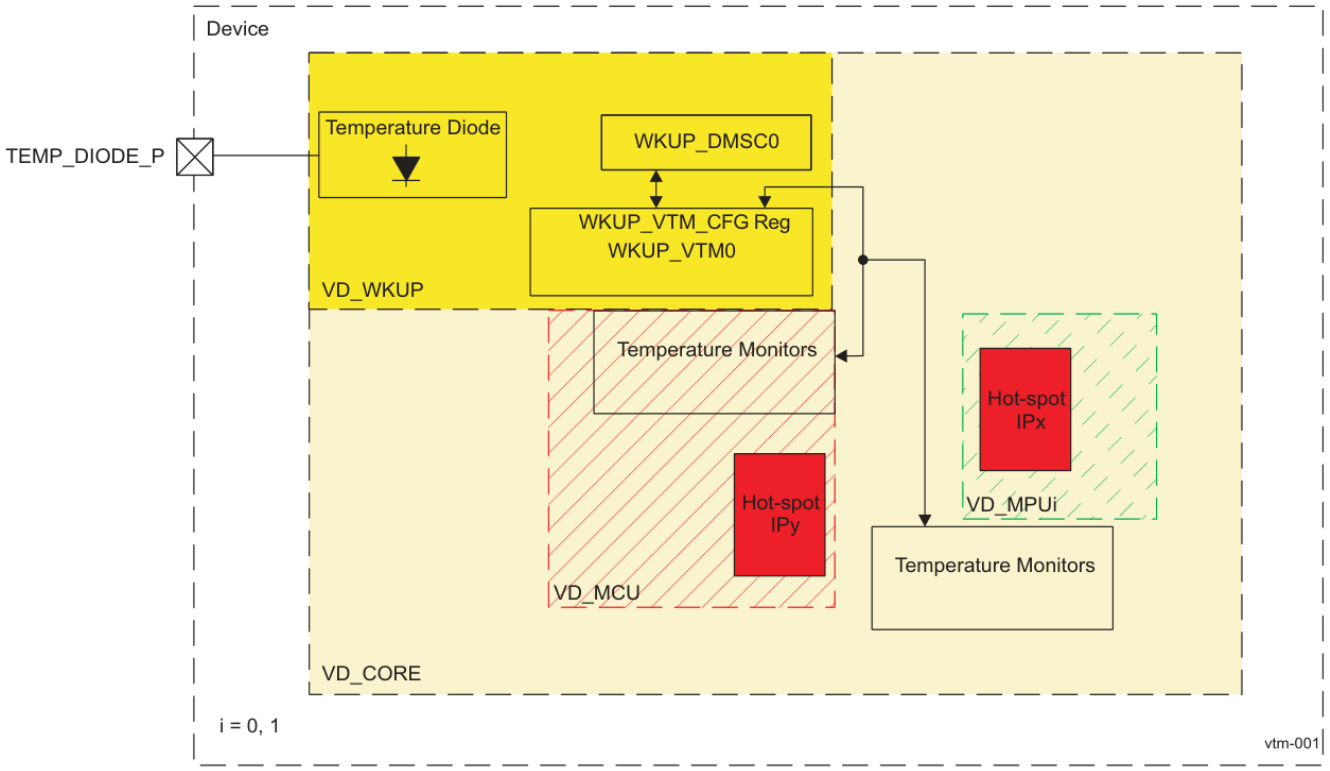

图 1-2. VTM 布局

不同的 SOC 通常需要不同数量的温度传感器，原理是尝试将传感器放置在靠近热源的位置，并覆盖所有热源。表 1-1 收集与其在所有 TI 处理器中的位置相关联的温度传感器。客户可以根据所应用的相应 SOC 来循环结果。该表根据热点的物理距离来进行分类。由于传感器主要位于两个热源及不同电源域边界之间，因此很难明确定义。该表格只能用作传感器位置的粗略参考。有关详细信息，请参阅 TRM（技术参考手册）。

表 1-1. 片上温度传感器

片上传感器	总数	A72/A53	DDR 控制器	C7x	R5F	GPU	编解码器	DPHY
TDA4VH	7	√	√	√	√	√	√	√
TDA4VE/VL/AL	7	√	√	√	√	√	√	√
TDA4VM	5	√	√	√	√	√	-	-
TDA4VEN	3	√	√	-	-	√	-	-
DRA821	3	√	√	-	√	-	-	-
AM62A	3	√	√	√	-	-	-	-
AM62P	3	√	√	-	-	√	-	-
AM62x	2	√	√	-	-	-	-	-
AM64/AM24	2	√	√	-	-	-	-	-
AM62L	1	√	-	-	-	-	-	-

所有 SOC 都在 Arm 内核周围放置了一个传感器，因为内核是 SOC 中的最热点。由于 DDR 负责整个 SOC 的数据吞吐量并高速运行，因此 DDR 存在显著的过热风险。因此，通常还需要将传感器放置在 DDR 控制器中进行监测。

1.2 VTM 工作原理及用法

上一节提到，每个 VTM 最多可以控制八个温度传感器，VTM 寄存器可以从温度传感器采集数据并存储到寄存器中。该寄存器命名为 WKUP_VTM_TMPSENS_STAT_j[9-0] DATA_OUT。温度值以 10 位二进制数的形式存储在相应寄存器中。表 1-2 列出了几个典型的实际温度值及其对应的 10 位温度值。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

表 1-2. 片上温度传感器寄存器值及实际值转换

实际温度	-40	-25	0	25	50	75	100	105	125	150
10 位十进制	28	77	164	260	366	485	620	648	773	949
10 位十六进制	01C	04D	0A4	104	16E	1E5	26C	288	305	3B5

通过使用上表查找方法，可以得出读取的值与实际温度值之间的粗略对应关系。此方法还可以用于寄存器中存储的温度数据与实际值之间的转换。虽然该表格查找只能提供粗略的温度范围，但要获得准确的温度值，必须使用以下公式进行计算。

$$y = 6.0373 \times 10^{-8} \times x^3 - 1.7058 \times 10^{-4} \times x^2 + 0.32512x - 49.002 - 9.2627 \times 10^{-12} \times x^4 \quad (1)$$

使用 Python 脚本绘制上述公式可以生成以下图像，这大约是一个单调递增的函数。该脚本允许输入从 SOC 读取的寄存器值，执行之后，该脚本会自动计算相应的准确温度值并对其进行批注。脚本可以在此处下载。

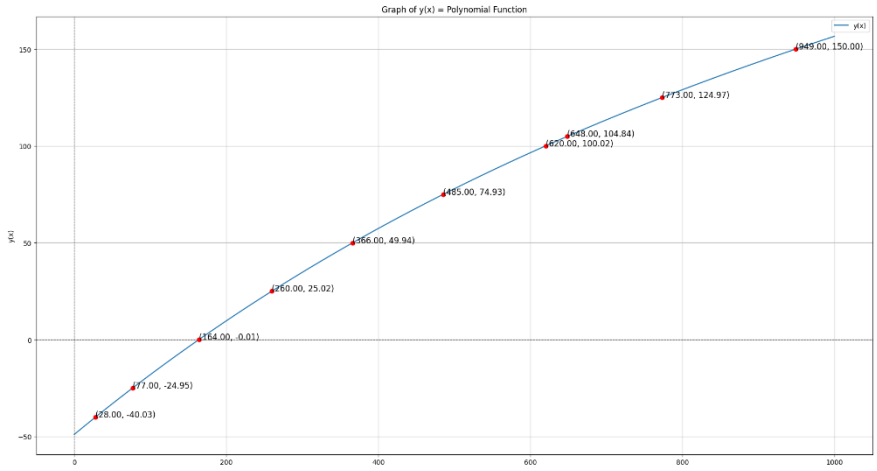

图 1-3. 寄存器值及实际值计算公式图

这可以精确计算相应的实际温度，从而更方便地修改系统温度保护逻辑。例如，修改 WKUP_VTM_MISC_CTRL2[25-16] MAXT_OUTRG_ALERT_THR0 和 WKUP_VTM_MISC_CTRL2[9-0] MAXT_OUTRG_ALERT_THR 可以更改 SOC 最高温度保护的阈值。

在 TI 提供的默认 Linux 软件中，相应的接口和命令可用于读取片上温度传感器的值。使用的命令类似。例如，在 AM62A EVM 上，可以使用以下命令读取三个温度传感器的值。所有温度传感器的值都以毫摄氏度为单位读取，结果显示如下：sensor1、sensor2 和 sensor3 显示的温度分别是 43.2°C、38.7°C 和 38.9°C。由于传感器位置的差异，测量的温度可能会发生变化，与平均值存在 ±5°C 的偏差为正常值。此范围在相应 SOC 的最新数据表中定义。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

```
root@am62axx-evm:/opt/edgeai-gst-apps# cat /sys/class/thermal/thermal_zone*/temp
43209
38749
38983
```

2 TI 处理器的硬件温度保护

前面几节主要介绍了 TI 处理器中负责温度管理的硬件温度管理模块。VTM 模块可从片上温度传感器读取值。获得温度后，VTM 根据测量的温度对整个 SOC 硬件执行热保护。

2.1 VTM 的过热保护阈值

VTM 的每个温度监测寄存器组均可配置为在其相应的温度监测器上执行温度采样，并触发最多 3 个警报信号。第一个是 10 位增量点 GT_TH1_ALERT（过热警报比较器 1 结果），该结果由阈值点 THPT1 生成。该值可以在 VTM_TMPSENSx_TH[25-16] TH1_VAL 中配置。第二个是 10 位增量点 GT_TH2_ALERT（过热警报比较器 2 结果），该结果由阈值点 THPT2 生成。该值可以在 VTM_TMPSENSx_TH2[9-0] TH2_VAL 中配置。第三个是 10 位增量点 LT_TH0_ALERT（过冷警报比较器结果），该结果由阈值点 THPT0 生成。该值可以在 VTM_TMPSENSx_TH[9-0] TH0_VAL 中配置。

通常，我们需要遵循 THPT2 > THPT1 > THPT0 的配置原理，工作原理在下文中进行了说明。

TH1 配置为早期警报，指示温度高于 VTM_VTM_TMPSENS_TH_j[25-16] TH1_VAL 定义的阈值。TH2 配置为警告，以指示温度高于 VTM_VTM_TMPSENS_TH2_j[9-0] TH2_VAL 定义的阈值。其概念是 TH1 表示处理器正在加热，TH2 需要立即引起注意。TH0 配置成当传感器检测到温度低于 VTM_VTM_TMPSENS_TH_j[9-0] TH0_VAL 定义的阈值时触发。该中断表示温度已降至原始 TH1 水平以下，从而允许放宽热缓解措施或退出紧急状态。请注意，如果启用了 LT_TH0_INT，则无论 TH1 和 TH2 中断是启用还是触发，当读取温度低于 TH0 时，都会触发 LT_TH0_INT。这三个中断可以由软件检测并应用于设计 SOC 热管理方案。客户可根据需求进行设计，这是通用参考逻辑。每个传感器都可独立设置过热或欠温阈值。只要一个传感器触发警告，就会生成相应的中断。如下图所示，所有传感器都可以为 TH0、TH1 和 TH2 生成与欠温和过热中断相对应的三个中断。客户软件之后可处理中断。请注意，中断仅在传感器处于连续模式时激活。单次采样不会触发任何中断。

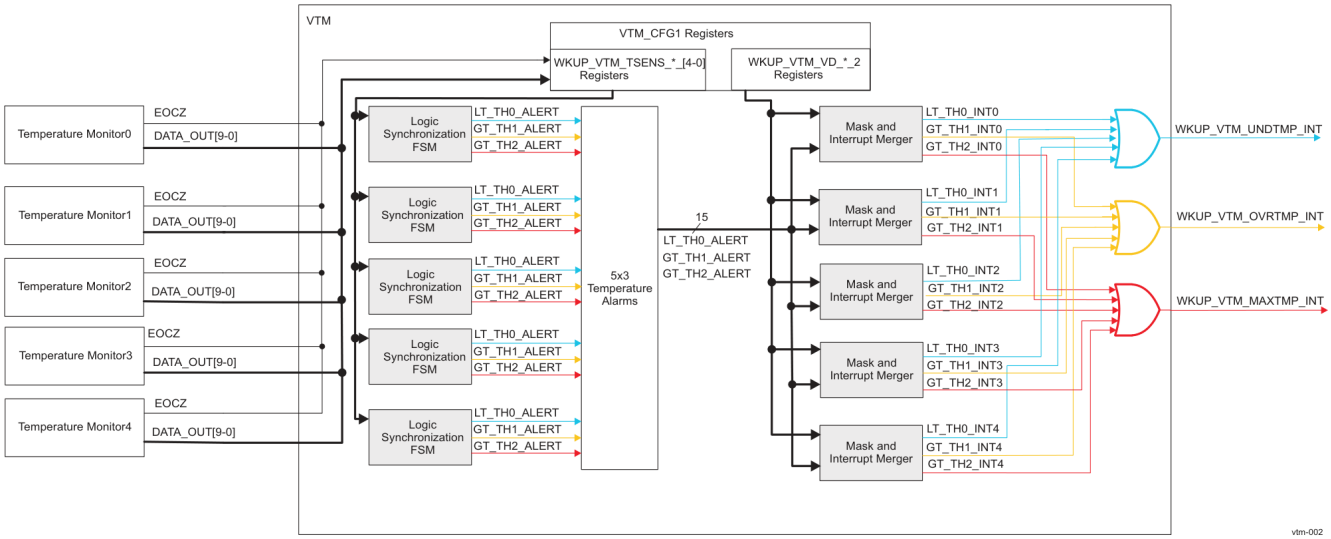

图 2-1. VTM 温度保护中断机制图

最终阈值为 MAXT_OUTRG_ALERT_THR。超过此阈值时，器件将被强制复位（并且将直接禁用 PLL）。一旦器件充分冷却（低于 MAXT_OUTRG_ALERT_THR0），内部复位将被释放，器件将重新启动。该温度保护基于硬件的保护机制。触发后，硬件将执行复位。客户可以修改相应的阈值以及是否启用触发器。然而，无法将该中断用于软件设计，因为 SOC 立即进入复位状态，从而无法进行任何软件操作。此外，这种保护不受传感器工作模式的限制。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

2.2 最高硬件温度保护

为了确保 SOC 在不超过其最大工作温度的情况下运行并保证其正常使用寿命，TI 处理器采用了基于硬件的最高温度保护机制。此机制不需要软件参与，在 TI SOC 上默认启用。本节介绍此保护的内部机制。为了让 TI SOC 在温度超过最大阈值时触发 VTM 警报，必须满足以下条件。

1. 为了确保 VTM 最高温度保护的有效输出，必须将 VTM 配置为启用至少一个温度传感器，具体通过设置 WKUP_VTM_TMPSENS_CTRL_j[11] MAXT_OUTRG_EN = 1 来实现。
2. 此外，位 WKUP_VTM_MISC_CTRL[0] ANYMAXT_OUTRG_ALERT_EN 必须设置为“1”。此位控制 VTM 最高温度保护的警告信号输出开关。
3. 在确保输入和输出路径的背景下，例如，在正确配置和启用温度传感器并启用最高温度保护输出后，如果芯片温度超过 WKUP_VTM_MISC_CTRL2[9-0] MAXT_OUTRG_ALERT_THR 中的编程值，则一旦传感器检测到编程的最高温度，VTM 输出将被驱动为“1”。
4. 在 SOC 级别，VTM 输出 THERM_MAXTEMP_OUTRANGE_ALERT，从而将 PLL 控制器驱动到热复位状态。同时，PLL 控制器进入复位状态，所有 PLL 同时进入旁路模式。
5. 根据第 4 点，该器件将在几秒钟内显著降低功耗并降低器件温度。
6. VTM 持续地监测温度。一旦 VTM 检测到与 WKUP_VTM_MISC_CTRL2[25-16] MAXT_OUTRG_ALERT_THR0, VTM 中的编程值相对应的代码，就会将 THERM_MAXTEMP_OUTRANGE_ALERT 驱动为 0。
7. 完成第 6 点后，PLL 控制器退出复位状态并重新启动引导序列。同时，PLL 退出旁路模式并以目标频率启动整个芯片，开始正常运行。

最高温度硬件保护流程图如图 2-2 所示

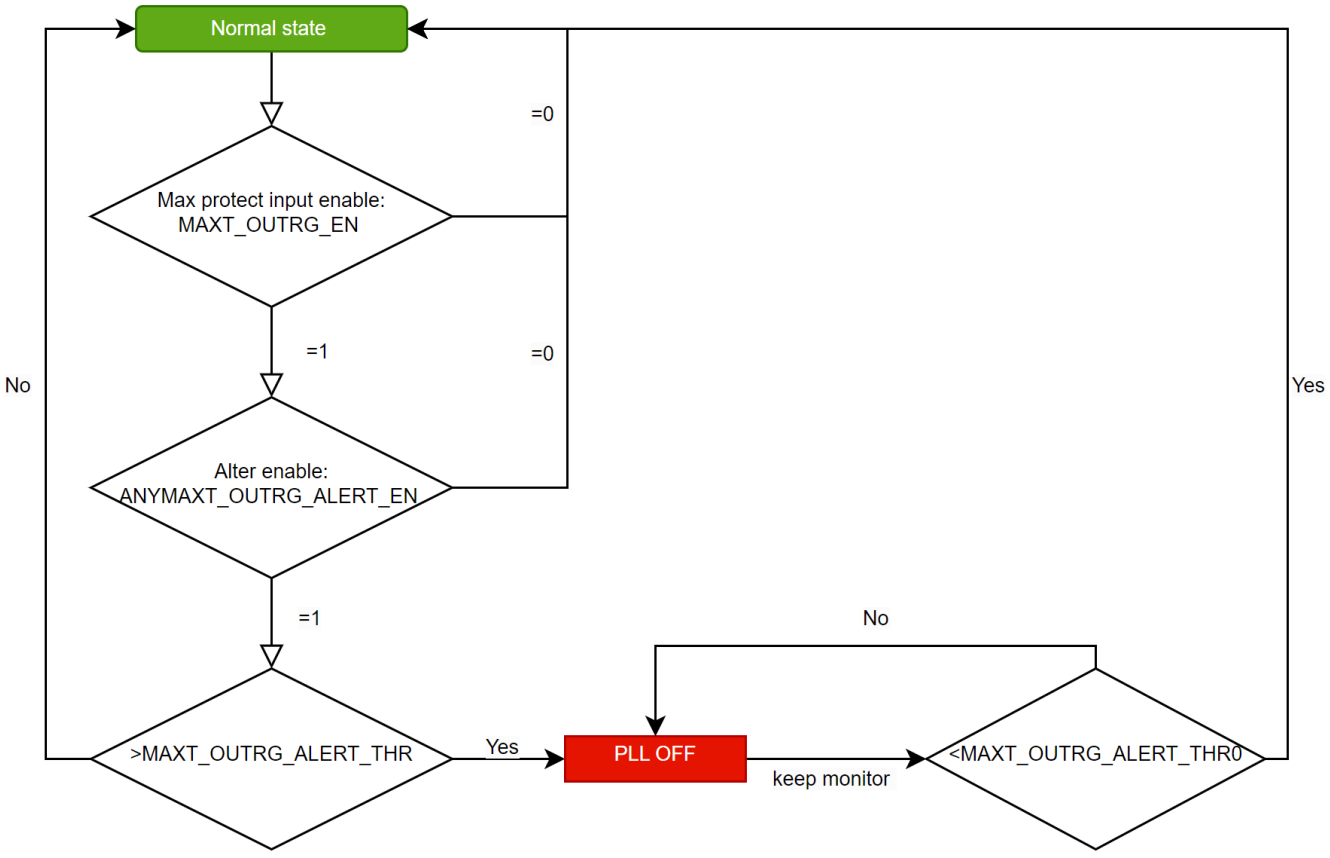

图 2-2. VTM 最高温度硬件保护图

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

在所有 TI SOC 系列中，默认启用最高温度保护。如果 SOC 超过最高温度，则会对 SOC 的使用寿命造成永久损坏，这种影响是不可逆转的。可以在相应的器件数据表中找到不同 SOC 的正常工作温度范围。在激活最高温度保护期间，SOC 的时钟 PLL 会进入旁路模式，从而有效地禁用 SOC 的整个时钟信号。此时，VTM 使用来自传感器的信号。

以下示例使用 AM62A 作为案例研究。我们可通过读取相应的寄存器以及 TRM 中详细说明的特定寄存器地址来获得 SOC 最高温度保护值。具体而言，WKUP_VTM_MISC_CTRL2[25-16] MAXT_OUTRG_ALERT_THR0 的编码值为 0x288，并且使用 Python 脚本，该值对应于 105°C 的实际温度。同样，WKUP_VTM_MISC_CTRL2[9-0] MAXT_OUTRG_ALERT_THR 的编码值为 0x2F8，对应于 123°C 的实际温度。因此，AM62A 的默认最大硬件温度保护值为 123°C。只有当温度降至 105°C 以下时，系统才会重新启动。

```
root@am62axx-evm:/opt/edgeai-gst-apps# devmem2 0x00b01010
/dev/mem opened.
Memory mapped at address 0xffff8488e000.
Read at address  0x00B01010 (0xffff8488e010): 0x028802F8
```


English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

3 软件温度保护策略

VTM 模块是一个硬件模块，主要管理整个 SOC 的温度。尽管 VTM 模块包含最高温度的硬件保护功能、灵活的温度保护阈值和过热保护中断，这些可供用户配置的设计使用，但基于硬件的温度保护却有限。除了在达到最高温度时停止 SOC 时钟之外，硬件模块无法直接控制其他热保护措施（例如降频）。此外，由于 TI 处理器是多核异构架构，并且每种处理器模型包含不同数量的温度传感器和首选工作温度，因此默认 VTM 温度保护阈值设置为 0，这意味着默认情况下仅启用最高温度保护。其他温度保护中断需要根据系统要求执行用户配置。因此，VTM 模块目前缺少除停止 SOC 之外的任何温度保护策略。作为补充，TI 添加了软件级温度保护策略。软件可以通过 VTM 模块获取温度值，并生成相应的中断或标志，从而使系统能够采取诸如降低频率、停止高负载运行和降低系统负载等措施。这对于系统功能安全而言至关重要。

3.1 可选软件温度保护措施

TI 处理器可提供相应器件的功耗估算电子表格，该电子表格基于测量数据和仿真数据来估算功耗。尽管 SOC 的工作功耗取决于电气参数、工艺变化、环境条件和处理器运行期间的用例等各种因素，但它仍可用作 SOC 功耗的粗略参考。因此，使用该工具作为一般参考，可以评估频率调节和降低负载等软件措施。SOC 的温度和功耗是正相关的，功耗越高、温度升高越快。因此，软件热管理的主要目标是降低 SOC 功耗。

表 3-1 以 AM62A 处理器为例，可从此处下载相应的工具。此方法也适用于其他 TI 处理器。在 125°C 结温下且所有内核均以最大频率工作时，对表 3-1 的结果进行了统计分析。

表 3-1. 不同负载下的 AM62A 功耗

案例	功耗/mW	A53	DDR	C7x	R5F	DM_R5	Δ Power/mW
正常状态	5136	80%	80%	80%	80%	80%	0
A53 负载下降	4936	20%	80%	80%	80%	80%	200
DDR 负载下降	4689	80%	20%	80%	80%	80%	447
C7x 负载下降	4181	80%	80%	20%	80%	80%	955
R5F 负载下降	5113	80%	80%	80%	20%	80%	23
DM 负载下降	5109	80%	80%	80%	80%	20%	27
所有负载下降	3127	20%	20%	20%	20%	20%	2009

表 3-2 中的结果是在 125°C 结温和所有内核均在 80% 负载下运行时得出的。

表 3-2. 不同工作频率之下的 AM62A 功耗

案例	功耗/mW	A53	DDR	C7x	R5F	DM_R5	Δ Power/mW
正常状态	5136	1400MHz	3200MHz	1000MHz	800MHz	800MHz	0
A53 频率下降	4837	700MHz	3200MHz	1000MHz	800MHz	800MHz	299
DDR 频率下降	5091	1400MHz	1600MHz	1000MHz	800MHz	800MHz	45
C7x 频率下降	4473	1400MHz	3200MHz	500MHz	800MHz	800MHz	663
R5F 频率下降	5109	1400MHz	3200MHz	1000MHz	400MHz	800MHz	27
DM 频率下降	5084	1400MHz	3200MHz	1000MHz	800MHz	400MHz	52
所有频率下降	4220	700MHz	1600MHz	500MHz	400MHz	400MHz	916

从计算结果可以看出，AM62A 中的 C7x 内核对功耗的影响最为显著。因此，在设计系统温度控制方案时，在达到目标温度后，应优先降低 C7x 内核的频率和负载。

Copyright © 2026 Texas Instruments Incorporated

3.2 Linux 温度保护逻辑

从 SDK 11.0（或更早版本）开始，Linux SDK 开始整合 Linux VTM 驱动程序。Linux 内核 VTM 及 SOC 的硬件 VTM 是两个不同的概念。内核 VTM 框架使用设备树配置（例如 k3-am62-thermal.dtsi，由内核热绑定文档定义）来监控传感器温度并执行相应的操作，例如降低 CPU 频率、关闭或重新启动 Linux。Linux 使用的传感器温度可以从 AM62x SOC 硬件 VTM 模块获得。

内核设备树 k3-am62a-thermal.dtsi 中的定义和配置仅为一个示例。客户可根据工程要求修改此设置，例如，用于关闭或重新启动 Linux。使用内核器件树 k3-am62a-thermal.dtsi 中的默认设置，内核会在温度达到 105°C 时触发关断序列。对于 Linux VTM 11.0，Linux SDK 驱动程序还为硬件 VTM 定义了类似的中断阈值，支持多达三个可编程温度阈值。其中两个阈值适用于高于阈值的值，一个适用于低于阈值的值，使 VTM 能够提醒内核采取措施。例如，当超过第一个阈值时，可以提醒内核开始降低 CPU 的电压和时钟速度，从而使 SoC 的整体温度稳定。如果 SoC 温度继续升高，我们可以使用第二个阈值来采取更积极的措施。例如，我们可以发出断电命令，以便在超过第二个阈值时完全关断器件。可使用设备树中定义的值在内核中设置阈值温度。这可用于设置内核关闭 SoC 的*临界*温度。当 SoC 过热时，它会向内核发送*被动*警报，可以通过将 cpufreq 驱动器注册为冷却器件来降低 MPU 频率。可以通过更改代码来修改软件热保护阈值，例如在 105°C 和 125°C 之间调节关断温度，这会将软件保护功能从工业级温度切换到汽车级温度。以下代码表示采用 2°C 迟滞控制的 95°C 的临界温度，在达到 95°C 时进入*无源*状态以启动冷却措施，并在 105°C（2°C 迟滞控制）时立即关断。

```
/* From arch/arm64/boot/dts/ti/k3-am62a-thermal.dtsi */
thermal_zones: thermal-zones {
     main0_thermal: main0-thermal {
          polling-delay-passive = <250>;      /* milliSeconds */
          polling-delay = <500>;           /* milliSeconds */
          thermal-sensors = <&wkup_vtm0 0>;
          trips {
               main0_alert: main0-alert {
                    temperature = <95000>;
                    hysteresis = <2000>;
                    type = "passive";
               };
               main0_crit: main0-crit {
                    temperature = <105000>;      /* milliCelsius */
                    hysteresis = <2000>;     /* milliCelsius */
                    type = "critical";
               };
          };
          cooling-maps {
               map0 {
                    trip = <&main0_alert>;
                    cooling-device =
                         <&cpu0 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>,
                         <&cpu1 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>,
                         <&cpu2 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>,
                         <&cpu3 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>;
               };
          };
     };
```

当内核进入*无源*状态时，内核可以通过消除低优先级进程来降低相应内核的负载。对于关键任务，降低工作频率可以降低功耗。动态频率选择 (DFS) 是动态调整 CPU 频率以优化性能的有效方法。有关更多详细信息，请参阅教程。

DFS 方法目前允许便利地调整 A 内核的频率。以下部分提供更改其他内核频率的方法。首先，确定与 TRM（时钟部分）中的内核相关联的时钟源、PLL 和分频器。根据 PLL 修改分频器，以确保最终的输出时钟符合速度等级。

以下是关于如何修改 AM62A 的 C7x 内核时钟的示例。PLL 被识别为 MAIN_PLL7 HSDIV0，可在相应器件的 TRM 中找到。所进行的更改如下：原始 1GHz 主时钟频率分频为 500MHz。可使用此方法对其他内核进行类似的频率调整。更改之后，可以在 Linux 中使用 k3conf 命令来验证相应的时钟频率。客户还可以使用 k3conf set clock $CLOCKID $FREQ 命令来修改时钟。可能需要修改不同器件的器件 ID；例如，AM62A 中的 C7x 对应于 ID 208 和 211。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

```
k3conf dump clock 208
k3conf dump clock 211
output:
|-----------------------------------------------------------------------------------------------|
| Device ID | Clock ID | Clock Name                          | Status          | Clock Frequency |
|-----------------------------------------------------------------------------------------------|
|   208     |     0    | DEV_C7X256V0_C7XV_CORE_0_C7XV_CLK | CLK_STATE_READY | 500000000   |
|-----------------------------------------------------------------------------------------------|
|   211     |     0    | DEV_C7X256V0_CLK_C7XV_CLK                       | CLK_STATE_READY |
500000000       |
|   211     |     7    | DEV_C7X256V0_CLK_PLL_CTRL_CLK                   | CLK_STATE_READY |
500000000       |
diff --git a/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
b/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
index 2122081..7438db5 100644
--- a/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
+++ b/source/drivers/device_manager/rm_pm_hal/rm_pm_hal_src/pm/soc/am62ax/clocks.c
@@ -2440,7 +2440,7 @@ static const struct clk_data_div_reg clk_data_hsdiv0_16fft_main_7_hsdiv0 = {
 static const struct clk_data_div_reg clk_data_hsdiv0_16fft_main_7_hsdiv0 = {
       .data_div          = {
            .n          = 128,
-           .default_div     = 2,
+           .default_div     = 4,
       },
       .reg               = 0x00680000UL + (0x1000UL * 7UL) + 0x80UL + (0x4UL * 0UL),
       .bit               = 0,
```

请注意，修改上述代码后，需要对 SDK 中的 lib 文件进行干净构建，并且由于修改后的代码在 DM（器件管理）内核上运行，执行与 DM 内核相对应的固件的干净构建。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

3.3 Linux 禁用没有使用的内核

TI 处理器由具有不同内核的异构架构组成（例如 TDA4VH），它包括 8 个 A72 内核、8 个 R5F 内核和 4 个 DSP C7x 内核。一些客户选择将 TDA4VH 专门用于 8 个 A72 和 R5F 内核，但不利用 4 个 DSP C7x 内核。由于 TI 在超集配置中为给定处理器提供开源 SDK，因此可提供完整规格的性能，并执行相应的软件修改以删除未使用的内核，从而减少不必要的功耗。以下示例演示了对 SDK 10.0 中 TDA4VH 里四个未使用的 DSP C7x 内核进行软件修改，以将其删除并降低实际工作温度。

```
diff --git a/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts b/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts
index de256005f..dff4c4408 100644
--- a/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts
+++ b/arch/arm64/boot/dts/ti/k3-j784s4-evm.dts
@@ -1310,28 +1310,28 @@
 };
 &c71_0 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster4 &mbox_c71_0>;
        memory-region = <&c71_0_dma_memory_region>,
                        <&c71_0_memory_region>;
 };
&c71_1 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster4 &mbox_c71_1>;
        memory-region = <&c71_1_dma_memory_region>,
                        <&c71_1_memory_region>;
 };
 &c71_2 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster5 &mbox_c71_2>;
        memory-region = <&c71_2_dma_memory_region>,
                        <&c71_2_memory_region>;
 };
 &c71_3 {
-       status = "okay";
+       status = "disabled";
        mboxes = <&mailbox0_cluster5 &mbox_c71_3>;
        memory-region = <&c71_3_dma_memory_region>,
                        <&c71_3_memory_region>;
 };
```


English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

4 总结

TDA4x 与 AM6x 是新一代 TI 处理器，包含功能强大的 VTM 温度管理模块。此模块可帮助 TI 的异构多核处理器变得高效而强大，使其能够在安全温度范围内以最大容量运行。处理器的处理能力越强，产生的热量越高。因此，客户必须在电路板的整体硬件设计阶段考虑元件的热设计。在实际系统运行期间，系统应始终在元件的指定工作温度范围内运行。本文中提到的温度保护措施是针对异常情况的补充措施，不应被视为常规热管理解决方案。VTM 仅支持片上温度的测量。为了在系统板上的其他位置（如 DDR 和 eMMC）进行温度管理，客户需要自行安装温度传感器或其他传感器。本应用手册介绍了 VTM 模块的工作机制，以及 TI SOC 上当前启用的相应硬件和软件温度保护策略。

5 参考资料

1. 德州仪器 (TI)，*AM62Ax Sitara™ 处理器*，数据表。
2. 德州仪器 (TI)，*TDA4VM 处理器*，数据表。
3. 德州仪器 (TI)，*AM62Ax Sitara 处理器技术参考手册*，技术参考手册。
4. *J721E DRA829/TDA4VM 处理器器件版本 2.0、1.1 技术参考手册*，技术参考手册
5. 德州仪器 (TI)，*(+) [常见问题解答] TDA4VM/TDA4VL/TDA4AL/TDA4VH/DRA821：我们如何使 Jacinto SDK 与各种器件兼容？YXZ - 处理器论坛 - 处理器 - TI E2E 支持论坛*，常见问题解答。
6. 德州仪器 (TI)，*(+) [常见问题解答] AM625/AM623/AM620-Q1/AM62Ax/AM62D-Q1/AM62Px/AM62L/AM64x/AM243x（ALV、ALX）定制电路板硬件设计 – 电压和热管理器 (VTM) – 处理器论坛 - 处理器 - TI E2E 支持论坛*，常见问题解答。
7. 德州仪器 (TI)，*(+) [FAQ] TDA4VM：如何使用 Linux 读取 J7 系列 SoC 上的片上温度*，常见问题解答。
8. 德州仪器 (TI)，*(+) AM625：Linux 热关机*，论坛。
9. 德州仪器 (TI)，*如何通过精确测温提升 CPU、GPU 和 SoC 性能*，常见问题解答。
10. 德州仪器 (TI)，*(+) [FAQ] SK-AM62：如何测量 AM62A 和 AM62X 的功耗和温度？*常见问题解答。

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

6 修订历史记录

Changes from Revision * (October 2025) to Revision A (May 2026)	Page
• 将 5°C 更改为 2°C	3
• 将 55°C 更改为 95°C	9
• 将 125°C 更改为 105°C	9
• 将 5°C 更改为 2°C	9
• 更新了整个文档中的表格、图和交叉参考的编号格式	12

English Document: SDAA146
Copyright © 2026 Texas Instruments Incorporated

重要通知和免责声明

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、与某特定用途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安全、安保法规或其他要求。

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 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 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。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

版权所有 © 2026，德州仪器 (TI) 公司

最后更新日期：2025 年 10 月