参考文献

Application Note

在 Linux 中利用动态频率调节实现 CPU 冷却

Keerthy J, Udit Kumar, Josiitaa RL

摘要

本应用手册介绍了德州仪器的 TDA4VL/J721S2 处理器如何借助基于 Linux 的动态频率调节 (DFS) 和热框架来实现高效的 CPU 热管理。本文介绍了通过 k3_j7xxx_bandgap 驱动程序可实现的当前热传感器支持，并概述了使用 DFS 启用被动冷却的步骤。本指南面向嵌入式 Linux 开发人员和系统架构师，适用于功率与热效率至关重要的汽车及工业市场应用场景。本文信息可帮助开发人员了解在基于 TDA4VL/J721S2 的系统中，如何监测热区并实现可扩展的软件控制冷却策略。本示例演示了在 TDA4VL 上的实现过程，且可扩展应用于其他 TDA4 及 Sitara 器件，以实现 Linux 系统下的热管理功能。

内容

商标

1 简介

1.1 TDA4VL SoC 概述

德州仪器的 TDA4VL SoC 隶属于 K3 系列，专为高性能汽车及工业应用设计。SoC 搭载双核 Arm®Cortex®- A72 CPU、可扩展存储器架构和集成加速器，兼具强大计算能力与高灵活性。在这类应用中，高效的功耗与热管理至关重要，尤其是在负载及环境条件多变的场景下运行高计算量 Linux 系统时。

1.2 文档目的

本应用手册介绍了如何在 TDA4VL 器件上利用基于 Linux 的动态频率调节 (DFS) 及热框架实现 CPU 功耗与温度的高效管理。本文重点介绍了当前热传感器的支持现状，探讨了 DFS 功能，并为将这些特性集成到嵌入式 Linux 系统中提供指导。

本文还概述了在 TDA4VL 平台上的实现示例，演示如何通过 Linux CPUFREQ 及热框架实现基于热状态的 CPU 频率调节。

1.3 目标受众与应用

本文档适用于：

- 嵌入式 Linux 开发人员
- 系统架构师
- 应用工程师
- 从事 TI TDA4VL 或其他基于 ARM 架构的 SoC 相关工作的工程师

本文档适用于以下应用场景：

- 汽车（ADAS、中央计算 ECU）
- 工业自动化和机器人

1.4 问题说明

在热受限环境中，CPU 持续保持高性能运行可能导致过热、系统降频或关机。DFS 与 Linux 热框架的集成，提供了一种基于软件的设计方案：可根据系统温度动态控制 CPU 频率，无需复杂的硬件冷却设计即可保障系统运行可靠性与效率。

2 Linux 中的动态频率调节 (DFS)

2.1 什么是 DFS？

动态频率调节（DFS，又称 CPU 频率调节）是一种电源管理技术，可使处理器根据性能需求或热状态动态调整工作频率。降低频率可减少功耗与发热量，而提高频率则能提升性能。

在德州仪器的 TDA4VL SoC（它搭载多个 Arm®Cortex®- A72 内核）上，DFS 可以用于通过在运行时调整 CPU 频率来平衡性能和功耗。这在需同时应对动态负载变化与热约束条件的嵌入式系统中，尤为实用。

2.2 Linux CPUFREQ 框架

Linux 内核提供了一个名为 CPUFREQ 框架的标准化接口，用于启用和控制 DFS。该框架允许用户和系统软件在运行时在硬件支持的不同 CPU 频率间切换。

CPUFREQ 框架的核心组件包括：

- CPU 频率驱动程序：这些驱动程序与特定硬件机制交互，以控制 CPU 时钟。
- 调节器：决定何时以及如何更改 CPU 频率的算法
- Sysfs 接口：允许用户空间与 CPUFREQ 子系统进行交互（例如，/sys/devices/system/cpu/cpu *cpufreq/）

2.3 支持的 CPUFREQ 调节器

Linux 内核为 DFS 提供了多种调节器，每种调节器适用于不同的使用场景：

- 性能调节器：
  - 始终将 CPU 设置为支持的最高频率。
  - 最适用于需持续保障最高性能的应用场景。
- 用户空间调节器：
  - 允许用户空间应用程序或脚本手动设置 CPU 频率。
- 按需调节器：
  - 根据当前 CPU 负载动态调整频率。
  - 常用于通用型系统。

每种调节器均可通过 sysfs 接口进行选择和配置，或由系统策略自动选择。

2.4 TI SoC 上的 DFS 支持现状

- AM62 器件：
  - 支持完整的 DFS 功能。
  - CPUFREQ 框架可根据所选调节器动态调节频率。
- TDA4 器件：
  - 截至当前版本 (SDK 11.0)，DFS 功能尚未启用。
  - 通过 k3_j7xxx_bandgap 驱动程序搭建的热监测基础架构已部署就位，但运行时频率调节功能尚待启用。
  - 一旦启用 DFS，基于 TDA4 的系统可在热应力下降低功耗与温度。

3 Linux 热框架

Linux 热框架是 Linux 内核中的子系统，用于监测并管理设备的热状态。该框架提供与平台无关的基础架构，支持定义热区、注册温度传感器，并应用热缓解策略（如动态频率调节 (DFS)、CPU 热插拔、风扇控制等）。

3.1 热区与触发点

热区指系统中需监测温度的区域，例如 CPU 集群、GPU 或 PMIC。每个热区都有以下关联项目：

- 一个或多个温度传感器
- 一组触发点，即阈值温度
- 达到触发点时会被激活的关联冷却设备

触发点用于定义动作阈值（例如，被动冷却或主动冷却）。触发点可分为以下类型：

- 被动：触发基于软件的缓解措施（例如降低 CPU 频率）
- 主动：触发硬件机制（例如启动风扇或提高风扇转速）
- 紧急：用于关闭系统以防止硬件损坏

这些触发点通常在设备树中定义，并由内核中的热调节器逻辑处理。

3.2 冷却机制：被动与主动

热框架支持多种冷却设备类型，大致可分：

被动冷却：

- 通过降低性能或功耗实现系统降温
- 示例：
  - 利用 DFS 降低 CPU 频率
  - 禁用或热插拔 CPU 内核
- 在需无噪音运行或低功耗运行的场景中，被动冷却为优选方案。

主动冷却：

- 使用外部机制实现降温，例如：
  - 打开风扇
  - 通过 PWM 控制来提高风扇速度
- 需外部硬件和驱动程序支持（如风扇控制器）

3.3 DFS 在被动冷却中的作用

在 TI TDA4VL 等平台上，DFS 可在 Linux 热框架下作为被动冷却设备使用。当某一热区温度超过设定的触发点时，热框架会请求降低 CPU 频率以实现降温。

该集成过程包括：

- 将 CPU 频率驱动程序注册为冷却设备
- 将 CPU 频率与对应热区关联
- 定义与冷却状态关联的触发点

在 TDA4VL 上启用 DFS 支持后，用户可实现：

- 随温度升高动态降低 CPU 频率
- 维持系统稳定性并防止过热
- 在热受限环境中运行，无需额外硬件冷却

4 TDA4VL 器件的热支持

德州仪器 TDA4VL 平台集成片内热传感硬件，并支持通过 Linux 热框架实现温度监测。尽管 A72 内核的完整动态频率调节 (DFS) 功能尚未启用，但基础热监测能力已具备。

4.1 VTM 与带隙传感器初始化

TDA4VL SoC 中的电压与温度监控器 (VTM) 子系统为热传感提供硬件支持。该子系统包括：

- 多个片内温度传感器（基于带隙原理）
- 用于监测热区的基础架构

在 Linux 系统中，VTM 硬件通过 k3_j7xxx_bandgap 驱动程序完成初始化。此驱动程序支持：

- 从 SoC 上的多个热传感器读取温度值
- 向 Linux 系统热框架暴露热区

初始化完成后，可通过以下 sysfs 接口访问温度数据： /sys/class/thermal/thermal_zone*/temp

4.2 通过 k3_j7xxx_bandgap 驱动程序实现温度监测

k3_j7xxx_bandgap 驱动程序负责热传感器的底层初始化，并提供用于注册热区的软件钩子。功能包括：

- 读取 SoC 中 A72 内核、GPU 及其他关键区域的实时温度。
- 将每个物理传感器映射到对应的逻辑热区。
- 与 Linux 热管理核心集成，以触发温度触发点

借助该功能，即使在主动/被动冷却机制尚未实现的情况下，开发人员也能观察系统热行为并规划冷却策略。

5 在 TDA4VL 上启用 CPU 冷却

本节提供了在 TDA4VL 器件上利用动态频率缩放 (DFS) 启用被动 CPU 冷却所需的补丁程序，并概述了验证该功能的步骤。

5.1 启用 CPU 冷却的补丁程序

```
From e9ffcedb3b567110fda18f4a49fb8e66672910e4 Mon Sep 17 00:00:00 2001
From: Keerthy <j-keerthy@ti.com>
Date: Fri, 20 Jun 2025 14:52:39 +0530
Subject: [PATCH] arch/arm64/boot/dts/ti/k3-j721s2-thermal.dtsi: Add CPUFREQ
 based cooling
Add CPUFREQ based cooling
Signed-off-by: Keerthy <j-keerthy@ti.com>
---
 arch/arm64/boot/dts/ti/k3-j721s2-thermal.dtsi | 14 ++++++++
 arch/arm64/boot/dts/ti/k3-j721s2.dtsi | 33 +++++++++++++++++++++
 2 files changed, 47 insertions(+)
diff --git a/arch/arm64/boot/dts/ti/k3-j721s2-thermal.dtsi b/arch/arm64/boot/dts/ti/k3-j721s2-
thermal.dtsi
index e3ef61c16..69727a5a5 100644
--- a/arch/arm64/boot/dts/ti/k3-j721s2-thermal.dtsi
+++ b/arch/arm64/boot/dts/ti/k3-j721s2-thermal.dtsi
@@ -39,12 +39,26 @@ main0_thermal: main0-thermal {
 thermal-sensors = <&wkup_vtm0 2>;
 trips {
+ main0_alert: main0-alert {
+ temperature = <45000>; /* millicelsius */
+ hysteresis = <2000>; /* millicelsius */
+ type = "passive";
+ };
+
 main0_crit: main0-crit {
 temperature = <125000>; /* milliCelsius */
 hysteresis = <2000>; /* milliCelsius */
 type = "critical";
 };
 };
+
+ cpu_cooling_maps: cooling-maps {
+ map0 {
+ trip = <&main0_alert>;
+ cooling-device =
+ <&cpu0 THERMAL_NO_LIMIT THERMAL_NO_LIMIT>;
+ };
+ };
 };
 main1_thermal: main1-thermal {
diff --git a/arch/arm64/boot/dts/ti/k3-j721s2.dtsi b/arch/arm64/boot/dts/ti/k3-j721s2.dtsi
index 4f416bb18..71dcb9e6f 100644
--- a/arch/arm64/boot/dts/ti/k3-j721s2.dtsi
+++ b/arch/arm64/boot/dts/ti/k3-j721s2.dtsi
@@ -51,6 +51,10 @@
 d-cache-line-size = <64>;
 d-cache-sets = <256>;
 next-level-cache = <&L2_0>;
+ clocks = <&k3_clks 202 0>;
+ clock-names = "cpu";
+ operating-points-v2 = <&cpu0_opp_table>;
+ #cooling-cells = <2>; /* min followed by max */
 };
 cpu1: cpu@1 {
@@ -65,6 +69,35 @@
 d-cache-line-size = <64>;
 d-cache-sets = <256>;
 next-level-cache = <&L2_0>;
+ clocks = <&k3_clks 203 0>;
+ clock-names = "cpu";
+ operating-points-v2 = <&cpu0_opp_table>;
+ #cooling-cells = <2>; /* min followed by max */
+ };
+ };
+
+ cpu0_opp_table: opp-table {
+ compatible = "operating-points-v2";
+ opp-shared;
```

```
+
+ opp6-2000000000 {
+ opp-hz = /bits/ 64 <2000000000>;
+ clock-latency-ns = <300000>;
+ };
+
+ opp4-1000000000 {
+ opp-hz = /bits/ 64 <1000000000>;
+ clock-latency-ns = <300000>;
+ };
+
+ opp2-500000000 {
+ opp-hz = /bits/ 64 <500000000>;
+ clock-latency-ns = <300000>;
+ };
+
+ opp1-250000000 {
+ opp-hz = /bits/ 64 <250000000>;
+ clock-latency-ns = <300000>;
 };
 };
--
2.17.1
```

5.2 测试 TDA4VL 上的冷却功能

应用补丁、重建并部署内核后，可按以下步骤验证 CPU 冷却功能的运行情况：

1. 验证冷却设备注册状态。

   ```
   cat /sys/class/thermal/cooling device*/
   ```

   查找诸如 cpu-freq 之类的条目，此类条目可表明 CPU 频率驱动已注册为冷却设备。
2. 检查热区触发点。

   ```
   cat /sys/class/thermal/thermal_zone1/trip_point_0_temp
   45000
   ```

   本示例中，触发点温度已设置为 45°C
3. 检查当前及最大状态。

   ```
   cat /sys/class/thermal/thermal_zone1/cooling_device0/cur_state
   0
   cat /sys/class/thermal/thermal_zone1/cooling_device0/max_state
   3
   ```

4. 检查当前及可用的调频范围。

   ```
   cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq
    2000000
   cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
    250000 500000 1000000 2000000
   ```

   当前 CPU 频率为 2GHz，可用频率范围为 250MHz 至 2GHz。
5. 检查当前温度。

   ```
   cat /sys/class/thermal/thermal_zone2/temp
   44753
   ```

   当前温度约为 44.753°C
6. 触发 CPU 负载。

   ```
   cpuloadgen 100 100 &
   [1] 1246
   ```

   使用 cpuloadgen 命令将 CPU 核心负载固定在 100% 并持续 100 秒，以提升 CPU 负载。

7. 检查当前温度。

```
cat /sys/class/thermal/thermal_zone2/temp
45209
```

随着 CPU 负载升高，SoC 温度升至 45.209°C，触发基于调频的 CPU 动态冷却功能。随后温度降至 44.981°C。

```
cat /sys/class/thermal/thermal_zone2/temp
44981
```

8. 检查当前调频频率及冷却状态。

```
cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_cur_freq
250000
cat /sys/class/thermal/thermal_zone1/cooling_device0/cur_state
3
```

当前 CPU 频率已降至 250MHz，设备已达到最大冷却状态。

备注

触发点温度被特意设定为 45°C，以此在室温环境下展示设备的完整功能。用户需根据系统对触发温度进行校准。

6 在 TDA4 及 Sitara 器件中的可扩展性

本应用手册中所述的在 TDA4VL 平台上启用基于 DFS 的 CPU 被动冷却的方法，可适配至其他具备类似 CPUFREQ 与热框架支持的德州仪器 TDA4 及 Sitara 系列 SoC。

6.1 调整实现方式

将该设计移植到 TDA4 或 Sitara 系列其他器件时，需执行以下操作：

1. 更新设备树节点
   - 根据目标 SoC 支持的频率调整 CPU OPP 表。
   - 根据可用测试条件配置热区
2. 验证 DFS 支持情况
   - 确认目标 SoC 的 CPUFREQ 驱动程序支持运行时频率变更。
   - 在内核配置中启用对应的调节器。
3. 热区映射
   - 将 CPU 冷却设备与新 SoC 的正确热区关联。
4. 带负载测试
   - 重复测试 TDA4VL 上的冷却功能中所述的功能测试程序，以验证冷却功能是否正常工作。

7 总结

本应用手册演示了在 TDA4VL 平台上如何通过 Linux 热框架实现基于 DFS 的 CPU 被动冷却。通过设备树配置将 CPUFREQ 散热设备与片内热传感器集成后，系统可动态调整 CPU 频率以维持安全工作温度，无需依赖主动散热组件。

本文档提供的补丁程序与测试流程，为开发人员在同类 TDA4 及 Sitara 系列器件上复现该设计提供了实用参考。只需对 CPU OPP 表、热区域配置及 CPUFREQ 调节器进行少量修改，该方案即可适配各类 TI SoC。

采用该方法不仅能提升系统可靠性与使用寿命，还能在热受限环境中实现无噪音、高能效的运行状态。

8 参考资料

1. Samsung Electronics Co,*CPU cooling APIs How To*
2. Intel Corporation, *Generic Thermal Sysf driver How To*
3. 德州仪器 (TI)，*TDA4VE TDA4AL TDA4VL Jacinto™ 处理器，器件版本 1.0* 数据表。

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。

最后更新日期：2025 年 10 月