Technical White Paper

选择合适的处理器和图形用户界面 (GUI)

Shriya Surti, Soobin Kim, Tobias Morell (KDAB)

摘要

选择合适的嵌入式用户界面 (UI) 框架既是技术决策，也是战略决策。本白皮书提供了面向平台的选择指南，映射了德州仪器 (TI) Sitara™ 处理器系列的主要 UI 框架。该指南将 Sitara 器件分为多个层，评估每个框架，并介绍了实施最佳实践。通过了解硬件限制和 UI 框架的交互方式，开发人员可以做出明智的选择，从而优化成本、性能和上市时间，满足其特定的产品要求。在进一步深入探讨之前，值得注意的是，没有单一的“最佳”处理器或 GUI 框架；正确的选择取决于每个项目的要求、团队专业知识和许可需求，并应随着硬件功能和生态系统的发展而得到重新审视。

内容

1 简介.....2
2 了解硬件现状及 UI 含义.....2
2.1 硬件平台类别：基线.....2
2.2 特色平台规范及架构环境.....3
3 技术深入研究：框架.....4
3.1 Qt：跨平台应用程序框架.....4
3.2 Slint：现代轻量化的替代品.....4
3.3 LVGL：微控制器专家.....5
3.4 Flutter：移动优先的方法.....6
3.5 其他值得注意的框架.....6
4 KDAB 决策框架和实施最佳做法.....7
4.1 超越器件的决策标准.....7
4.2 实施最佳实践：优化性能.....8
4.3 启动时间注意事项（“冷启动”问题）.....8
4.4 功效（电池供电设备）.....8
4.5 未来趋势和考虑事项.....8
5 总结：TI 和 KDAB 的观点.....9
5.1 关于 KDAB Group.....9
5.2 德州仪器 (TI) Sitara MPU Group 简介.....9
6 参考资料.....9

商标

Sitara™ is a trademark of Texas Instruments.
Arm® and Cortex® are registered trademarks of Arm Limited.
所有商标均为其各自所有者的财产。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

1 简介

为嵌入式系统选择合适的用户界面 (UI) 技术不仅是软件工程决策，也是战略性业务选择。无论最终应用如何，最终用户都需要 60fps 的动画、高分辨率显示和直观的触摸交互。

要实现这种性能水平，需要仔细考虑硬件功能、性能目标、开发资源和长期维护。对硬件进行过度设计以支持繁重的 UI 框架，这会增加物料清单 (BOM) 预算。相反，为功能强大的硬件选择基本的 UI 框架会使 GPU 资源没有被利用，并由于开发人员工具不当而延长上市时间。

本指南概述了先进的嵌入式 UI 技术，并将这些选项映射到 TI Sitara 系列的硬件范围内。

2 了解硬件现状及 UI 含义

在评估软件框架之前，了解硬件基础非常重要。嵌入式硬件频谱非常广泛，微控制器 (MCU) 和微处理器 (MPU) 之间的界限越来越模糊。硬件平台仍分为不同的类别，每个类别都施加了实质性的限制，从而影响了 UI 技术的选择。

2.1 硬件平台类别：基线

表 2-1. TI Sitara 处理器 HMI 层

TI Sitara HMI 层	器件	CPU	3D GPU / 视频	NPU	显示	UI 约束
高层	AM67A	多达四核 Cortex-A53	3D GPU 高达 50GFLOPS 和视频编解码器	多达 4 个 TOPS	三显示器支持	能够提供丰富的多显示 UI,并具有实际的带宽预期；根据器件的不同，可以实现 AI 集成和更高的功率预算。
	AM67			不适用		
	AM62P			不适用		
中层	AM62	多达四核 Cortex-A53	3D GPU 高达 8GFLOPS	不适用	双显示支持	具备入门级 3D 硬件加速功能的良好 CPU/RAM 余量。AI 集成可以根据器件提供。
	AM62A		不适用	多达 2 个 TOPS	单显示支持	
低层	AM62L	多达双核 Cortex-A53	不适用	不适用	单显示支持	部署在高度成本敏感的产品中，其中软件效率对于保持较低的 BOM 至关重要。
	AM335	单核 Cortex-A8	3D GPU 高达 1.6GFLOPS	不适用	单显示支持	

备注

适用性取决于实际内存配置、图形驱动程序、Linux/RTOS 栈、显示拓扑、应用复杂性以及在目标硬件上测量的行为。此表是选择指南，而不是支持或性能保证。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

2.2 特色平台规范及架构环境

为了上下文化 UI 框架在现实世界中的行为，它有助于检查当今最流行的嵌入式器件的特定架构。

备注

从第一天起，这些 SoC 中的许多都很少作为定制芯片设计进行部署。相反，它们通过来自行业先进企业的强大模块上系统 (SoM) 生态系统来提供产品。SoM 合作伙伴可以缩短现代物联网 (IoT) 器件的工业产品上市时间并且简化硬件认证。

2.2.1 德州仪器 (TI) AM335 平台

- **架构：**采用 NEON SIMD 技术的单核 1GHz Cortex-A8。所有应用程序代码都在同一个内核上运行，因此 UI 和任何实时处理会共享资源；没有专用的实时内核。
- **图形和显示：**Imagination SGX530 3D 图形和 24 位 LCD 控制器，支持高达 1920 × 1080 (30fps) 的单 RGB 显示。

2.2.2 德州仪器 (TI) AM62L 平台

- **架构：**双核 64 位 Cortex-A53（高达 1.25GHz），为 Linux UI 提供足够的处理能力，同时保持系统简单，具有低成本和低功耗功能。
- **图形和显示：**60fps 时支持高达 1920 × 1080 的 DSI 或者 DPI 单显示。

2.2.3 德州仪器 (TI) AM62A 平台

- **架构：**频率高达 1.4GHz 的四核 Cortex-A53，两个用于 MCU 通道和器件管理工作的 800MHz Cortex-R5F 内核和一个高达 2TOPS 的 C7x 深度学习加速器。R5F 内核处理确定性低延迟任务，而 A53 集群运行 Linux 和图形 UI。
- **图形和显示：**60fps 时支持高达 2048 × 1080 DPI 的单显示。

2.2.4 德州仪器 (TI) AM62 平台

- **架构：**四核 Cortex-A53（高达 1.4GHz）和单个 Cortex-M4F（高达 400MHz），支持客户在通用应用中运行响应迅速且多功能的 GUI。
- **图形和显示：**具有 2D/3D 加速功能的 Imagination AXE-1-16M GPU；双 LVDS/DPI 显示支持高达 1920 × 1080 (60fps)。

2.2.5 德州仪器 (TI) AM62P 和 AM67 平台

- **架构：**四核 Cortex-A53，频率高达 1.4GHz；800MHz 处有两个独立的 Cortex-R5F 内核。R5F 内核可以独立于 Linux 驱动的 UI 运行实时控制或者安全任务。
- **图形和显示：**具有 2D/3D 加速功能的 Imagination BXS-4-64 GPU；支持 LVDS、DSI 或者 DPI 高达 3840 × 1080 (60fps) 的三路显示。这些器件还具备 H.264/H.265 4K UHD 视频功能。

2.2.6 德州仪器 (TI) AM67A 平台

- **架构：**四核 Cortex-A53，频率高达 1.4GHz；800MHz 处有两个独立的 Cortex-R5F 内核。该器件包含两个深度学习加速器，速度高达 4TOPS，以支持 AI 功能。
- **图形和显示：**具有 2D/3D 加速功能的 Imagination BXS-4-64 GPU；支持 LVDS、DSI 或者 DPI 高达 3840 × 1080 (60fps) 的三路显示。此器件还具有 H.264/H.265 4K UHD 视频功能。

备注

KDAB 的观点：总体拥有成本方程式

在架构审查中，我们经常看到团队仅根据开发人员的熟悉程度选择 UI 框架（例如 Web 技术或 Flutter），但最终发现框架需要大量内存才能顺利运行。从低成本处理器配置升级到高端配置以吸收软件开销有可能会破坏产品的利润。必须以符号方式选择硬件及 UI 软件。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

3 技术深入研究：框架

3.1 Qt：跨平台应用程序框架

Qt 是一个完整的基于 C++的应用程序平台 — 不仅仅是一个 UI 渲染器 — 拥有超过 25 年的持续开发经验。Qt 为具有 10 至 15 年生命周期的产品提供工具、跨平台部署及长期商业支持 (LTS)。

3.1.1 技术架构

- **Qt Quick (QML)：**硬件加速场景图上的声明性 UI。
- **Qt Widgets：**传统的必备 C++ UI 工具包；适用于数据密集型接口或者 CPU 渲染接口。
- **Qt 快速编译器：**QML 提前编译，可缩短启动时间并减少存储器占用空间。
- **Qt Quick 3D：**3D 内容在 2D 界面内的原生集成。
- **Qt for MCUs (Qt Quick Ultralite)：**面向微控制器的定制 QML 子集和确定性运行时。

3.1.2 平台适用性和现实世界扩展

- **低层 Sitara 器件（AM335、AM62L）：**通过严格的优化和适当范围的 UX 设计，实现可行性。更大限度地减少 QML 嵌套深度，减少过度绘制，并尽早验证图形路径。Qt Widgets 可用于狭窄的数据导向型接口，但现代触摸 UX 需要大量的产品特定设计投资。
- **中端 Sitara 器件（AM62、AM62A）：**Qt Quick 的强大目标。这些平台为中等复杂的应用程序提供了 CPU、RAM 与显卡功能的平衡。测量目标硬件上的启动、存储器和持续帧行为；Qt Quick Compiler 可以减少启动时间工作。
- **高端 Sitara 器件（AM62P、AM67、AM67A）**：Qt 是一个理想的选择，具有硬件加速的图形路径和适用于丰富多显示器 UI 设计的范围。确认生产图形堆栈并且验证目标显示配置的组成和带宽。

3.1.3 商业案例：优势和挑战

- **优势：**成熟的生态系统；Qt Creator 及 Qt Design Studio（带 Figma 网桥）；在 Linux、RTOS、Windows、iOS 和 Android 上进行单代码库部署；通过 Qt Safe Renderer（一个单独的产品；这些认证不包括完整的 Qt 框架）为 IEC 62304 和 ISO 26262 提供认证支持；Squish 测试自动化。
- **挑战：**占用空间大而框架很少；围绕 CMake/qmake 的学习曲线；闭源产品或静态链接方案的许可复杂性。商业 Qt 许可能作为主要的总拥有成本考虑因素，具体取决于部署模型、产品数量、支持需求、静态链接方法和其他 Qt 产品。团队必须获得当前的商业条款，并评估许可与工程和 BOM 成本。

备注

KDAB 的观点：许可与开发速度

商业 Qt 许可是直接且持续的成本。真正的比较是在工具生态系统之间进行：Qt Creator 和 Design Studio，Slint 的实时预览工具以及 Flutter 的热重新加载可加速不同的工作流程。团队必须根据自己的产品来评估工具的适用性，而不是假设任何一个框架都具有独特的生产效率。

3.2 Slint：现代轻量化的替代品

Slint 是一个基于 Rust 的声明性 UI 框架，从一开始就为极高的效率、内存安全和开发人员的工作效率而设计。Slint 报告的运行时 RAM 占用空间低于 300KB，同时仍支持动画、触摸输入和硬件加速渲染。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

3.2.1 技术架构

- **声明性 UI 语言：**一种具备深度工具支持的自定义、高度可读标记语言。
- **多渲染后端：**适用于 MCU 的软件渲染器，外加 OpenGL、Metal、Vulkan、Direct3D 以及用于支持 MPU 的 Skia 加速渲染器。
- **语言绑定：**一流的 Rust 和 C++集成，与 Python 和 JavaScript/TypeScript 的附加绑定。

3.2.2 平台适用性和现实世界扩展

- **低端 Sitara 器件（AM335、AM62L）：**Slint 非常适合注重低占用空间、可预测存储器行为和声明性工作流程的应用。选择软件或加速渲染后端必须遵循生产图形堆栈及测量的工作负载。
- **中端 Sitara 器件（AM62、AM62A）：**该层与 Slint 的优势非常吻合。GPU 加速后端可在支持的情况下使用；根据实际图形 API 及驱动程序可用性规划渲染器。
- **高端 Sitara 器件（AM62P、AM67、AM67A）：**Slint 可以支持丰富、响应灵敏的加速 UI，包括多显示器设计。随着更多的资源，关键优势转移到 Rust 内存安全、许可模式和跨语言绑定。

3.2.3 商业案例：优势和挑战

- **优点：**大约 300KB 的运行时、Rust 级内存安全、实时预览和热重新加载，以及 Figma 导入，如果您已经拥有 Figma 设计（尽管它不是主要的创作工作流），它将非常有用。双 GPLv3 + 商业许可证非常简单。Slint 具有低 CPU 唤醒事件，因此非常适合使用电池供电的设备。
- **挑战：**Slint 是一个更年轻的框架。第三方小工具和生态系统深度跟踪 Qt。核心设计工具非常出色，但针对复杂边缘案例的 IDE 集成和全面文档仍在成熟、认证域跟踪记录仍处于初级阶段。

备注

KDAB 的观点：嵌入式中的 Rust 革命

作为 Slint 的服务合作伙伴，KDAB 会密切监控采用情况。我们看到 EV 充电及 IoT 工业器件团队越来越感兴趣。这些项目共享紧凑的内存和 CPU 预算、长久的使用寿命以及对软件安全的关键需求 – 这些领域是 Rust 大显身手的地方。Slint 允许这些团队在传统上仅支持基本、低分辨率 UI 的硬件上提供流畅动画界面。

3.3 LVGL：微控制器专家

轻量级多功能图形库 (LVGL) 是领先的开源 UI 框架，专门针对微控制器进行了优化。LVGL 完全用 C 语言编写，外部依赖性极小，在过去仅支持段或单色位图显示的硬件上实现了复杂的图形界面。

3.3.1 技术架构

- **纯 C 实现：**跨编译器和平台实现最大程度的可移植性。
- **硬件抽象化：**可插拔显示、输入及文件系统驱动程序。
- **渲染敏捷性：**高度优化的 CPU 软件渲染，具有用于特定于平台的 GPU 加速的挂钩。
- **RTOS 集成：**与 FreeRTOS、Zephyr、RT-Thread 或者裸机完全兼容。

3.3.2 平台适用性和现实世界扩展

- **低层 Sitara 器件（AM335、AM62L）：**LVGL 非常适合成本敏感型应用。LVGL 占用空间小，可为应用程序逻辑保留资源，是受控设备式接口的理想选择。
- **中端 Sitara 器件 (AM62、AM62A)：**当便携性、小尺寸或现有投资至关重要时，LVGL 仍然适用。对于图形丰富的 HMI 设计，评估面向 GPU 的声明性框架是否可以更有效地使用平台并提高迭代速度。
- **高端 Sitara 器件（AM62P、AM67、AM67A）：**在技术上，LVGL 是可行的，但当产品旨在广泛使用图形和多显示功能时，其价值可能会降低。这种选择必须是深思熟虑的，而不是自动的。

3.3.3 商业案例：优势和挑战

- **优势：**真正的 MCU 优化，商业用途不受限制的许可 MIT 许可证、活跃的社区以及用于视觉设计的 LVGL Pro。
- **挑战：**手动 C UI 代码比声明性方法更慢，更容易出错。Pure C 缺乏 Rust 的内存安全保证，给开发人员带来了正确分配和指针处理的负担。随着状态复杂性增加，必须使用的 API 变得很难维护。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

3.4 Flutter：移动优先的方法

Flutter 是 Google 对跨平台开发的愿景。Flutter 在移动开发人员中被广泛采用（在开发人员调查中始终在顶级跨平台移动框架中排名）。Flutter 的主要重点仍然是移动和 Web；嵌入式 Linux 支持功能强大，但以社区为导向，而不是完全受支持的第一方工作。

3.4.1 技术架构

- **Dart 语言：**垃圾收集，采用 JIT（开发、热重新加载）和 AOT（生产）编译。
- **自定义渲染：**通过硬件加速的 Skia 引擎绘制的所有像素（过渡到叶轮），确保像素的完美一致性。
- **小部件组成：**一种具有高度反应性更新的“一切即小部件”理念。

3.4.2 平台适用性和现实世界扩展

- **低端 Sitara 器件（AM335、AM62L）：**不推荐默认使用 Flutter。生产决策必须遵循包括启动、内存使用、持续响应、图形驱动程序行为及功率预算在内的目标板概念验证。
- **中端 Sitara 器件（AM62、AM62A）：**在产品具有足够 RAM 和成熟加速图形路径的位置，可以评估 Flutter。如果移动代码共享和可用的 Flutter 技能抵消了额外的运行时间成本，则 Flutter 是合理的。结果因工作负载和硬件配置而存在很大差异。
- **高端 Sitara 器件（AM62P、AM67、AM67A）：**如果系统提供足够的 RAM 和强大的 GPU 加速，Flutter 是一个视觉丰富应用的可靠选择。60fps 对于相应的工作负载是可实现的，但必须测量最终应用、图形堆栈、分辨率和显示拓扑。

3.4.3 商业案例：优势和挑战

- **优势：**无与伦比的开发人员体验：热重新加载、IDE 支持、丰富的内置小工具 (Material/Cupertino) 及大规模软件包生态系统 (pub.dev)。在移动配套应用程序和嵌入式设备之间共享代码库的功能减少了招聘瓶颈。
- **挑战：**严重的资源开销可能会迫使硬件 BOM 升级。嵌入式 Linux 主要由社区驱动：著名的嵌入器包括 Toyota 的 ivi-homescreen (https://github.com/toyota-connected/ivi-homescreen)，用于基于 AGL 的汽车信息娱乐系统，以及用于 Raspberry Pi 和类似主板的 Flutter-pi (https://github.com/ardera/flutter-pi)。Dart 的垃圾收集带来了不可预测的延迟，使 Flutter 不适合严苛的实时要求。在具有专用实时处理器或软件组件的系统中，这些任务可在架构上与 Flutter 驱动的 UI 分离。

备注

KDAB 的观点：Flutter 的“隐藏成本”陷阱

由于 Flutter 获得 BSD 许可，因此组织通常认为 Flutter 是最便宜的路由。但是，如果您的 C/C++ 团队必须学习 Dart，并且硬件团队必须将 RAM 翻倍并升级 GPU 以处理 Flutter 的开销，则 TCO 通常会超过 Qt 或 Slint 等高度优化框架的商业许可证成本。

3.5 其他值得注意的框架

- **Altia**：具有合作伙伴生态系统（包括德州仪器 (TI) 演示）的商用嵌入式和汽车 HMI 开发平台。对于寻求基于模型的 UI 工作流和供应商支持的嵌入式目标上部署的团队来说，这是值得评估的。权衡因素包括许可、生成的代码工作流、运行时占用空间、图形堆栈集成以及与现有应用程序架构的适配。
- **Avalonia UI：跨平台**：跨平台 .NET 框架 (C# / XAML)。可通过 DRM/KMS 在 Linux 上运行。权衡：.NET 运行时开销；不适合 MCU。
- **启动软件：**一种涵盖 MPU 和 MCU 的商用 GUI 工具链，旨在提供广泛的器件支持和快速 UI 原型设计；对于想要一个与供应商无关的设计/运行时管道的团队来说，这是值得评估的。
- **嵌入式向导**：商业 MCU/MPU 框架在汽车和白色家电领域非常强大。权衡：较高的许可成本。
- **Kanzi (Rightware)**：在汽车信息娱乐系统中占据主导地位的高级框架。权衡：高级许可。
- **.NET MAUI**：Microsoft 的 Xamarin 后继产品。强大的移动/桌面功能；嵌入式 Linux 支持仅限于社区。权衡：比嵌入式设备本身更适合作为配套应用。
- **Unreal Engine UMG**：用于高端标牌和豪华 EV 仪表板的 AAA 游戏引擎。权衡：繁重的硬件需求、陡峭的学习曲线、大型二进制文件。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

4 KDAB 决策框架和实施最佳做法

节 2.1 中的德州仪器 (TI) Sitara HMI 层级表是此版本的主要平台优先选择指南。表 2-1 提供要评估的主要起点和替代框架。

4.1 超越器件的决策标准

4.1.1 许可和业务模式

- **开源许可：**LVGL (MIT)、Flutter (BSD)。
- **开源通用公共许可证：**Qt (GPL/LGPL)、Slint (GPL)。回顾静态链接及动态化的含义。
- **商业要求：**Qt（闭源/静态）、Slint（商业）、Qt for MCUs、Embedded Wizard、Kanzi。

4.1.2 团队专业知识和开发速度

- **C/C++ 嵌入式团队：**Qt、Slint（C++ 绑定）或 LVGL。
- **Rust 团队：**Slint。
- **C# / .NET 团队：**Avalonia。
- **移动开发人员：**Flutter（硬件允许）。

备注

速度说明：声明性框架（QML、Slint、Flutter、XAML）产生的迭代速度通常比强制性手动编码（LVGL、Qt Widget）快得多。

4.1.3 长期维护及认证

- **已建立跟踪记录：**Qt（25 年以上）、LVGL（10 年以上）、Embedded Wizard。
- **新兴：**Slint（较新，商业支持）、Flutter 嵌入式（社区驱动）。
- **安全关键型 (IEC 62304/ISO 26262)：**Qt（商业）和 Embedded Wizard 是保守的选择。验证 Slint 是可能的，但需要投资。Flutter 的垃圾回收使硬实时认证极有挑战性。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

4.2 实施最佳实践：优化性能

4.2.1 内存和图形管理

- **确定性内存：**在可能的情况下进行静态分配；尽早配置最坏情况下的 RAM；经常创建及销毁池对象。
- **提前在目标硬件上测试：**一个常见且昂贵的陷阱是在工作站上开发整个 UI，并且仅在启动前不久在目标上进行验证。从第一个冲刺开始在目标硅上运行，包括散热测量和引导时测量。
- **资产优化：**压缩和流式传输资产。在 Qt 项目上，请始终使用 Qt 快速编译器。
- **渲染效率：**通过有效的图层组合最大程度减少过度绘制。使用 OpenGL、OpenGL ES、Vulkan 或者平台原生 API（如果可用）进行硬件加速。

4.2.2 功率效率

- 任何在不使用时可以被关闭的设备都必须关闭 — 包括 GPU、显示控制器、传感器和外设。结合系统的电源管理器来设计 UI 状态机。

4.3 启动时间注意事项（“冷启动”问题）

应用启动时间会严重影响嵌入式器件的感知质量。

- **Qt：**Qt 快速编译器通过 AOT 编译将冷启动减少数百毫秒。
- **Slint：**由于其编译的二进制文件大小很小，因此启动开销很小。
- **LVGL：**快速初始化，运行时设置很少。
- **Flutter：**由于 Dart VM 和框架加载，初始化时间更长。

通用优化：修整后的依赖项和预编译有助于快速启动。静态链接会有所帮助，但不是一个银色项目符号—有选择地应用它，而不是作为默认性能控制杆。

4.4 功效（电池供电设备）

- 根据 UI 活动实施自适应帧速率（例如，60fps 活动、30fps 空闲、1fps 静态）。
- 任何在不使用时可以被关闭的设备都必须关闭 — 包括 GPU、显示控制器、传感器和外设。结合系统的电源管理器来设计 UI 状态机。
- 批处理显示更新可降低存储器带宽并最大限度地减少 CPU 唤醒事件。

4.5 未来趋势和考虑事项

嵌入式 UI 环境持续快速发展。定期重新评估可以确保您的技术选择在较长的产品生命周期内保持最佳状态。

- **WebAssembly 和基于 Web 的 UI：**通过 WebAssembly 部署用于仪表板及远程监控的 UI 的兴趣日益增加。Qt for WebAssembly 和 Slint for WebAssembly 都是可行途径。权衡仍然是可移植性与运行时开销。
- **机器学习集成：**现代平台越来越多地包含 AI 加速器，并且 UI 框架与 ML 流水线集成，用于手势识别和预测界面。Qt 已经提供 ML 框架绑定。
- **生锈生态系统的增长：**内存安全无垃圾收集对于嵌入式系统非常有吸引力。Slint 作为一个 Rust 原生 UI 框架，而 Qt 正在积极研究试验 Rust 绑定。
- **硬件发展：**新 MCU 正在接近入门级 MPU 性能，模糊 MCU/MPU 边界，并在传统 MCU 预算中实现更丰富的 UI。Vulkan 的应用正在增长，有望在不同供应商之间实现更一致的图形 API。

English Document: SDAA554
Copyright © 2026 Texas Instruments Incorporated

5 总结：TI 和 KDAB 的观点

选择嵌入式 UI 技术是一种妥协。没有单一的“最佳”框架：只有特定硬件功能、团队专业知识、许可状态和产品生命周期组合的最佳框架。同样，没有一款“合适”的处理器：TI Sitara MPU 产品系列经过精心设计，可覆盖从成本优化的入门级设计到功能丰富的多显示和 AI 加速系统的完整嵌入式 HMI 范围。

在整个 Sitara MPU 产品系列中，TI 提供一致的软件基础，包括 Linux SDK 支持，显示驱动程序和硬件加速图形栈。这使得开发团队可以随着产品要求的发展，在不同层级之间切换，而无需丢弃软件投资。随着硬件能力的提高和生态系统的成熟，不适合当今项目限制的框架可能变得可行，因此将这一选择视为需要重新审视的决定，而不是一次性选择。

5.1 关于 KDAB Group

KDAB Group 是全球领先的软件咨询公司，负责 Qt、C++ 和 OpenGL 应用程序在台式机、嵌入式和移动平台上的架构、开发和设计。KDAB 是 Qt 最大的独立贡献者，也是全球首家通过 ISO 9001 认证的 Qt 咨询和开发公司。我们的专家构建运行时、混合本机和 Web 技术，并为数百个客户 (其中许多是财富 500 强客户) 解决硬件堆栈性能问题和移植问题。KDAB 在创建、调试、分析和移植复杂应用程序方面的工具和丰富经验可帮助全球开发人员成功交付项目。KDAB 的培训师，都是全职的开发人员，为 Qt、OpenGL 和现代 C++ 提供多种语言的市场领先的实践培训。

5.2 德州仪器 (TI) Sitara MPU Group 简介

德州仪器 (TI) 的 Sitara 产品系列是专为工业自动化、HMI、机器人、电机控制、边缘 AI 应用等 Sitara MPU 系列基于 Arm® Cortex®-A 架构构建，涵盖从入门级成本优化型器件到多核和 AI 加速器件的各种性能层。TI 提供全面的开发资源，包括适用于 Linux 的 Processor SDK、实时软件、参考设计、评估模块，以及广泛的模块上系统 (SoM) 生态系统和载板合作伙伴，可加快产品上市速度。凭借在嵌入式处理领域数十年的经验，TI 支持十年或更长的产品生命周期，可让工业和商业客户放心在 Sitara 上进行严苛、长寿命的部署。

6 参考资料

- KDAB Group，www.kdab.com
- 德州仪器 (TI)，www.ti.com

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