白皮书

在设计中使用 Arm® TrustZone® 的益处

2020 年 11 月

概要

从电池供电的无线温度传感器，到高功率工业电机控制应用，安全性对于任何电子设备制造商而言都是一个重要且影响深远的话题。确保最终产品的安全，不只是防止攻击者获得密钥和密码的访问权限，还包括防范应用的软件知识产权被复制，遭到入侵的设备还会被用作控制和利用其他系统的平台。

数字安全的本质涉及使用加密密钥和加密算法来保护对各个系统资源与功能的访问。这些功能需要处理器资源才能工作，而且这些重要的安全组件必须与应用中那些能够被其他系统访问的组件隔离开来。

隔离是安全性的一个重要层面，但对于如今功能丰富且高度集成的处理器而言，实现隔离是一项挑战。

本白皮书将介绍能通过系统级的 Arm® TrustZone® 来实现安全性，以及在 Renesas Advanced (RA) 32 位微控制器中 TrustZone 的具体实施。瑞萨电子 RA 产品家族基于 32 位 Arm® Cortex®-M 微控制器架构，提供四个不同的系列，这些系列均在低功耗和高性能之间进行了优化。

对隔离的需求

如今，很少有嵌入式系统不与其他设备和应用互相连接。无论是有线还是无线连接，互联互通无处不在，物联网 (IoT) 设备极易受到攻击。实现物联网的前提是设备能够访问本地网关或直接访问云端资源。例如，位于制造工厂内部的环境感知边缘节点定期连接到本地或基于云的相关资源，以传输温度、湿度及其他环境参数进行分析。该数据经流程控制应用解读后，可能向工厂的供暖与通风系统边缘节点发送启动指令。在这两种应用场景中，边缘节点应用将访问加密数据和密钥，来进行身份验证并加密或解密通信。如果应用以这种方式处理数据，同时其他应用代码也在同一个处理器上运行且能够窥视外设接口，则边缘节点将很容易受到攻击者的攻击。

访问及使用安全流程和密钥的应用代码需要与不涉及安全数据的其他代码段完全隔离。此外，在最低的物理层的隔离至关重要，包括在系统的引导过程，因为许多漏洞攻击会专门针对设备的启动过程。

Arm TrustZone 技术提供高效的嵌入式机制,将单核设备的硬件域分成两个不同的隔离环境。一个环境用于运行任意代码，另一个环境运行那些需要完全隔离、分离和安全区域的代码。请参见图 1。

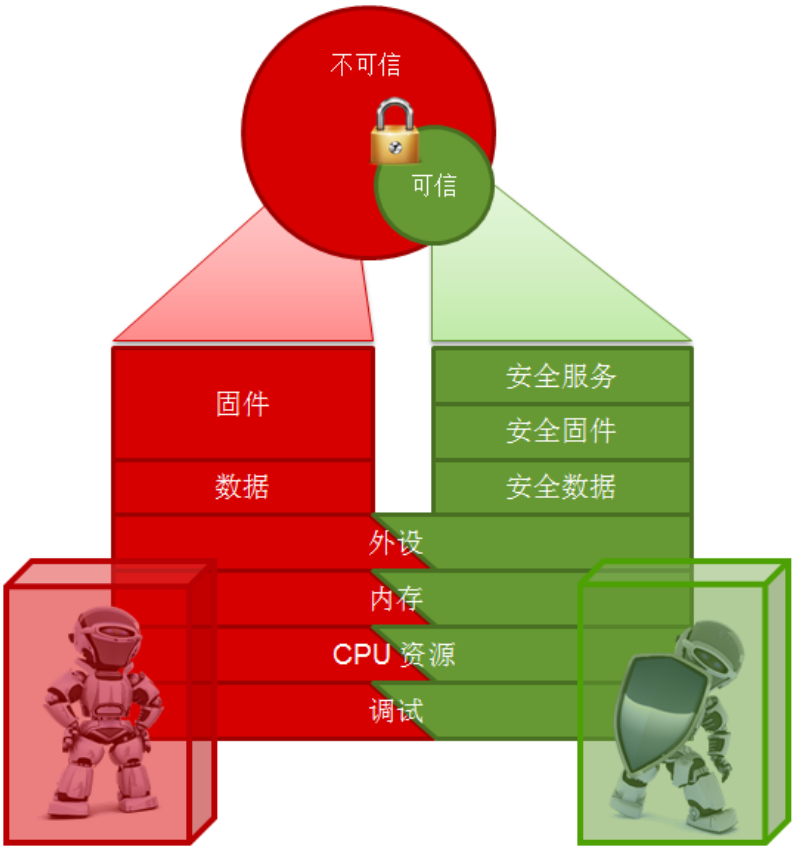


图 1：Arm TrustZone 在硬件层面强制隔离微控制器的功能

介绍 Arm TrustZone

Arm TrustZone 已问世十余年，首次出现是作为安全扩展引入 Cortex®-A 级应用处理器。在 Cortex-A 微处理器上，这两个不同且隔离的环境被称为安全环境或可信环境和非安全环境或不可信环境，通常运行各自独立的操作系统实例。直到最近，TrustZone 才可用于 Cortex-M 微控制器内核系列。与微处理器相比，微控制器会有一些资源限制，因此 M 系列 MCU 上的 TrustZone 实现稍有不同，主要是为了在性能和电源效率方面开销更低。并不是所有微控制器供应商都实现了 TrustZone，软件资源和工具的生态系统仍在开发中。

Arm TrustZone 是可信执行环境（TEE）的一个示例，在安全环境和非安全环境之间隔离数据、服务以及特定内存区域。在安全区域内运行的包括私钥、固件和安全例程等，十分灵活；当然也可能也包含商业敏感型库或算法，这些是重要的 IP，需要保护。TrustZone 可阻止非安全代码访问安全区域内的软件和资源。安全区域可访问其区域内的所有资源。负责管理这两个环境之间的访问权限的是非安全可执行跳板（Non-secure callable veneer）。请参见图 2。跳板提供预定义的接入点机制，使非安全代码可以调用安全区域内的服务。

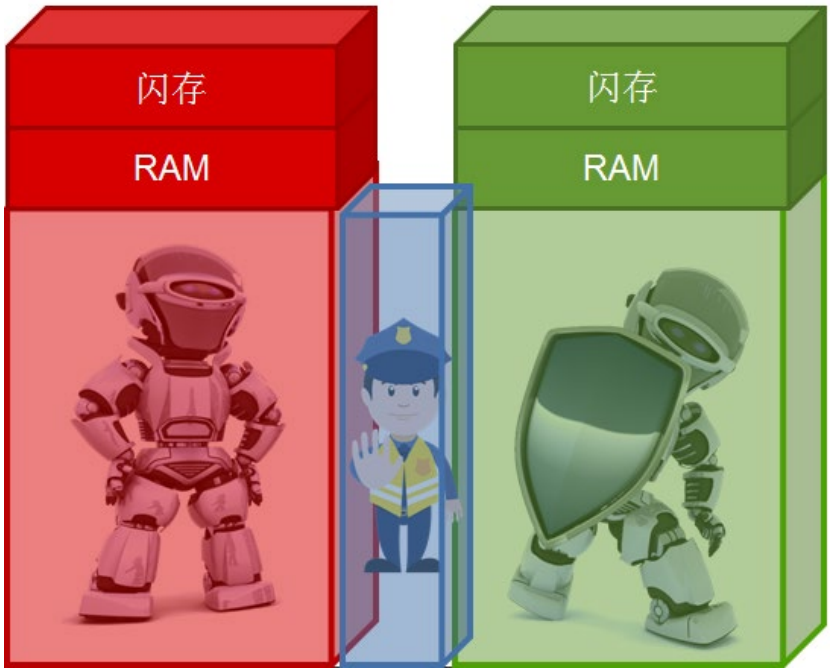


图 2：通过非安全可调用跳板提供的功能，非安全环境可访问安全环境的服务

TrustZone 提供的分离特性显著减少了攻击者可见的关键组件的攻击面，简化了对嵌入式设备的安全评估。Arm 信任的实现方式不仅包括软件封装，还扩展至闪存和 RAM；但是，Arm 对 TrustZone 的原始定义并不适合直接内存访问控制器 (DMA) 或直接传输控制器 (DTC) 请求。此外，在 Arm 的 TrustZone 定义中，外设和引脚访问的分离方式不同于应用代码和数据。因此，暴露的通用端口(GPIO) 和外设功能会导致微控制器容易遭受各种漏洞攻击。

瑞萨 RA 产品家族中 Cortex-M33 内核 MCU 的 Arm TrustZone

如上所述，瑞萨 RA 产品家族中 Cortex-M33 微控制器的分离和隔离架构以 Arm 的 v8-M TrustZone 为构建基础，但有几项重大改进。请参见图 3。客户工程团队完全可以选择是否在 RA6M4 中使用 TrustZone；然而，在我们的互联世界中，强烈推荐使用这些功能安全地部署物联网端点和边缘节点，以获得信心和保证。瑞萨认识到，为所有应用提供有弹性且可靠的安全功能的需求不断增长，于是对 Arm TrustZone 的实现进行了进一步改进，提供安全的分离和隔离机制，当下即可满足嵌入式系统的未来需求。

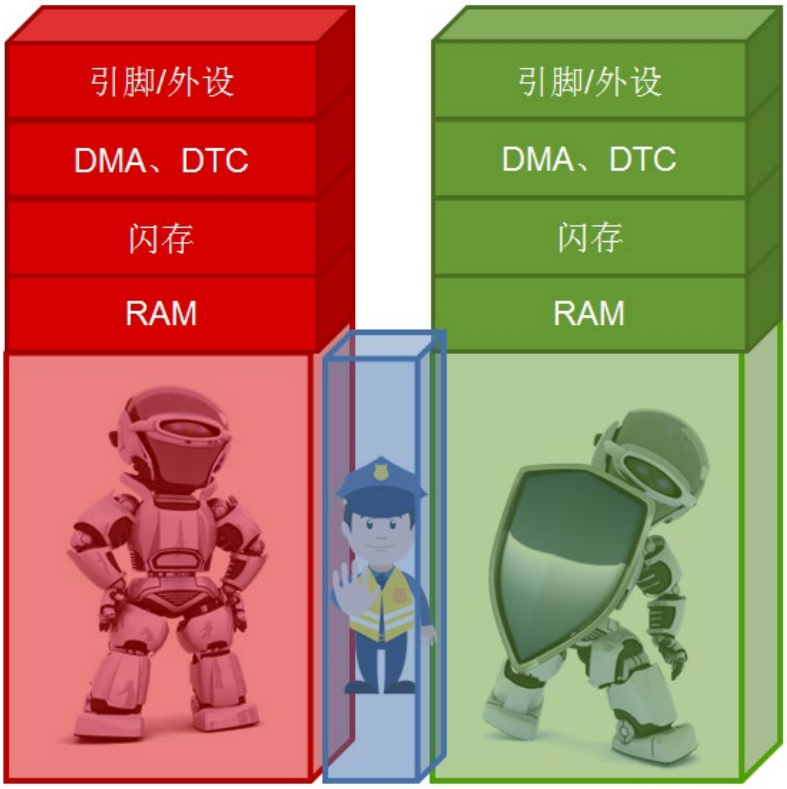


图 3：瑞萨在 RA 产品家族中 Cortex-M33 微控制器上实现 Arm TrustZone v8-M

RA 产品家族中 Cortex-M33 MCU 实现的 Arm TrustZone 可防止非安全代码使用 DMA、DTC 及其他内存访问机制来提取安全代码和数据。此外，RA6M4 的 TrustZone 筛选器应用于所有外设和引脚，以保护所有外部接口，防止非安全代码进行覆盖输出，并禁止非安全代码窃取输入信号。

从内存保护单元向 Trustzone 发展

在 Arm TrustZone 可用于微控制器之前，瑞萨意识到安全对互联设备的重要性，并创新实现了安全内存保护单元 (MPU)。对于未内置隔离与分离功能的处理器，可以通过安全 MPU 增强安全性；然而，尽管安全 MPU 为设计架构增添了新的特性，但多种应用场景证实了有必要进一步进行系统隔离。瑞萨通过在 RA 产品线 中实现 Arm TrustZone，增强了了瑞萨 RA 系列的隔离与安全功能。请参见图 4。

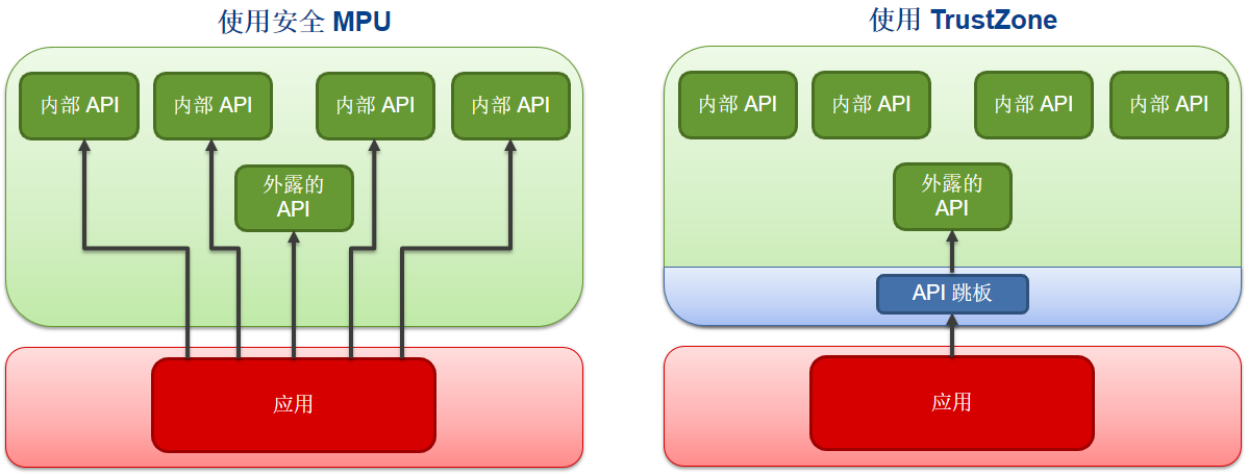


图 4：隔离 API，避免其暴露。使用安全内存保护单元与使用带有 Arm TrustZone 的瑞萨 RA6M4 的比较

通过安全 MPU，逻辑上应用应该访问 API，但无法限制它们的访问权限。考虑这样一种情况，有文档描述的外露 API 可能包含特定的安全功能，或提供对商业敏感型算法的访问权限。如图 4 所述，通过安全 MPU，对内部 API 的访问方式没有限制。只要您知道 API 的入口地址以及调用参数 A，便可以使用这些 API。如果您发现内部 API 中的某个特别有用的子例程可访问，并且也有参数可供使用，仍可使用非安全代码进行访问，尽管这不是推荐的访问方式；但是，使用 TrustZone 的情况下，通过外露的 API 访问内部 API 是通过硬件强制执行的。非安全应用代码只能访问非安全可调用空间中的 API 跳板。没有任何机制可以让非安全应用代码直接访问安全环境的内部 API。

另一个类似示例是防止代码滥用。请参见图 5。

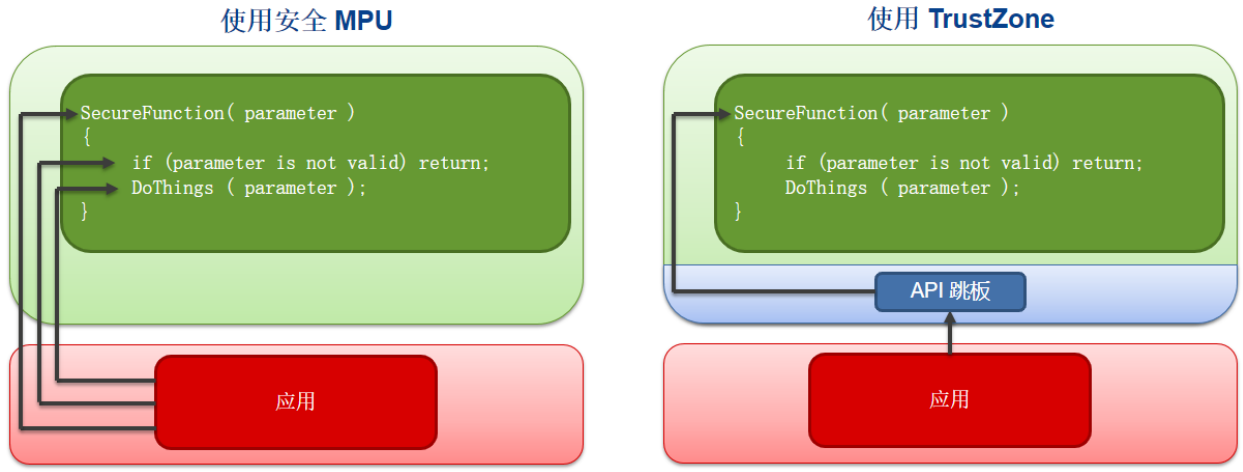


图 5：通过 TrustZone 防止代码滥用

使用安全 MPU，非安全访问可跳转至特定地址，从而绕过真正的入口。这种方式的代码滥用不受限制，无论是无意还是有意的。如果开发人员意识到，非安全代码跳转至带有某个功能的特定点而非顶部或许能够带来一些性能或操作优势，则安全 MPU 对此毫无办法。使用 TrustZone 时，以正确方式访问安全代码是硬件强制执行的。非安全代码只能通过单一入口点 API 跳板，访问安全代码顶部。该方法可阻止任何试图使用其他物理地址调用安全代码的操作。

瑞萨实现的 TrustZone 也禁止不受限制地访问外部接口和引脚。禁止非安全代码直接读取或写入引脚寄存器，这样做可避免窃取信号和端口模拟。

通过限制应用对 API 的访问、防止代码滥用和限制外部接口，减少了攻击者可用的潜在攻击面。

应用用例

Arm TrustZone 在非安全环境和安全环境之间提供的隔离功能适合多个不同的用例。接下来将分别探讨三个用例：IP 保护、代码分离和保护信任根。

IP 保护

考虑到创建专业算法（如电机控制算法）所牵扯的时间与精力，算法开发人员希望保护其 IP 不被复制的愿望是可以理解的。有一种方法既能实现这一愿望，同时仍维持应用灵活性，那就是算法开发人员将算法预先编程到微控制器的安全区域内，并提供 API 让应用开发人员可以访问算法。请参见图 6。

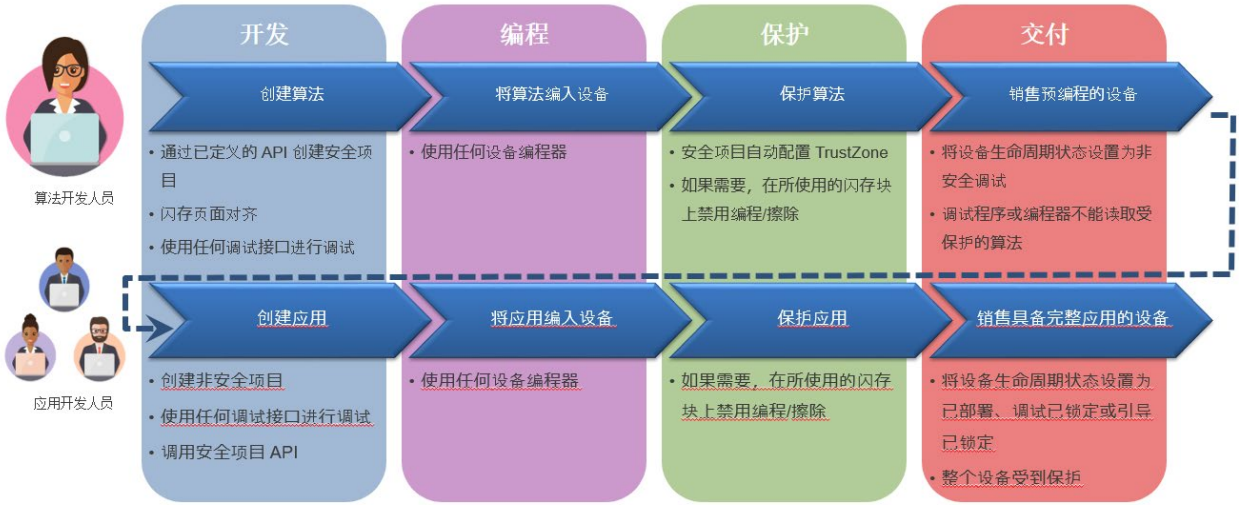


图 6：使用 TrustZone 的预编程算法 IP 保护的应用用例

将算法写入安全区域后，算法内存区域的编程和擦除功能便被禁用。然后，算法开发人员就可以销售这些预编程的设备供应用开发人员立即使用应用开发人员使用非安全代码进行编程，非安全代码通过非安全可调用跳板 来调用算法函数。算法完全受到保护，避免未授权访问和侵权。

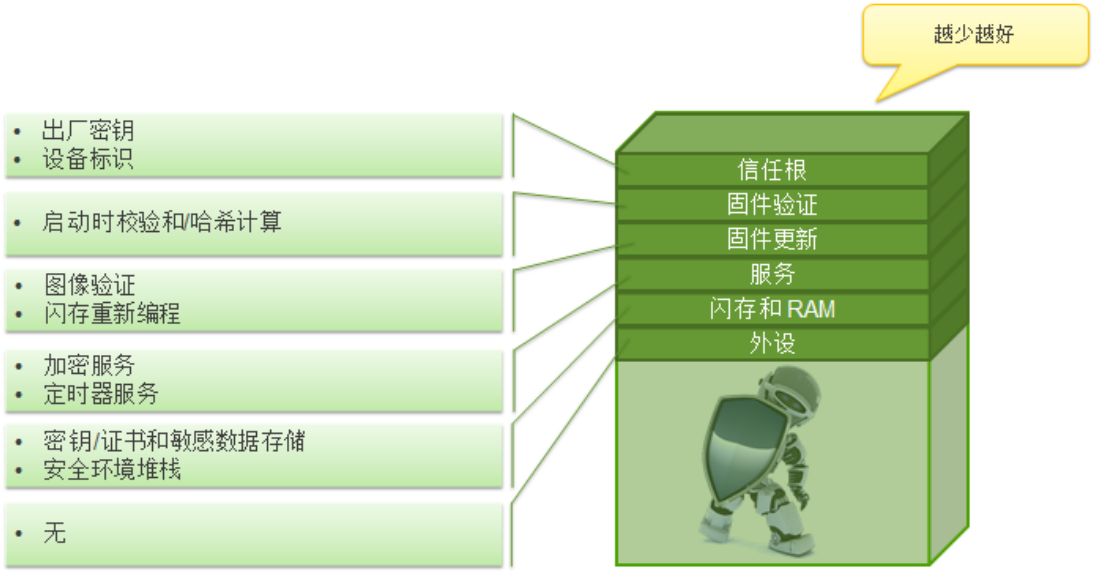
代码分离

欧洲智能电表规格要求是 TrustZone 代码分离用例的一个很好的例子。在测量行业指令 (MID) 规格中，用于测量和账单计算的应用代码的各个方面必须得到认证。该代码被认定为法律相关代码，必须与智能电表应用的其余部分隔离开来。目前，只有通过以物理方式分离两个微控制器，才能实现这一目的；然而，虽然这种方法或许能够简化认证流程，但使用两个 MCU 会产生一笔额外的材料成本，提高功耗预算，占用更多 PCB 空间。

另一个方法是使用 Arm TrustZone 提供的逻辑分离。所有经认证的法律相关代码可放置在安全区域中，不需要认证的其余代码可放置在非安全区域内。

保护信任根

设备的信任根 (RoT) 提供安全性的基石，这是一切的构建基础。RoT 包括出厂设置的密钥、经过身份验证的固件、设备标识和启动时校验和计算所需的凭据。如果设备因高级安全故障而遭到入侵，则可以通过 RoT 恢复；然而，如果 RoT 被攻破，则基于其构建的所有安全性将不再可信。强烈建议将所有信任根元素保存在安全环境中。请参见图 7。



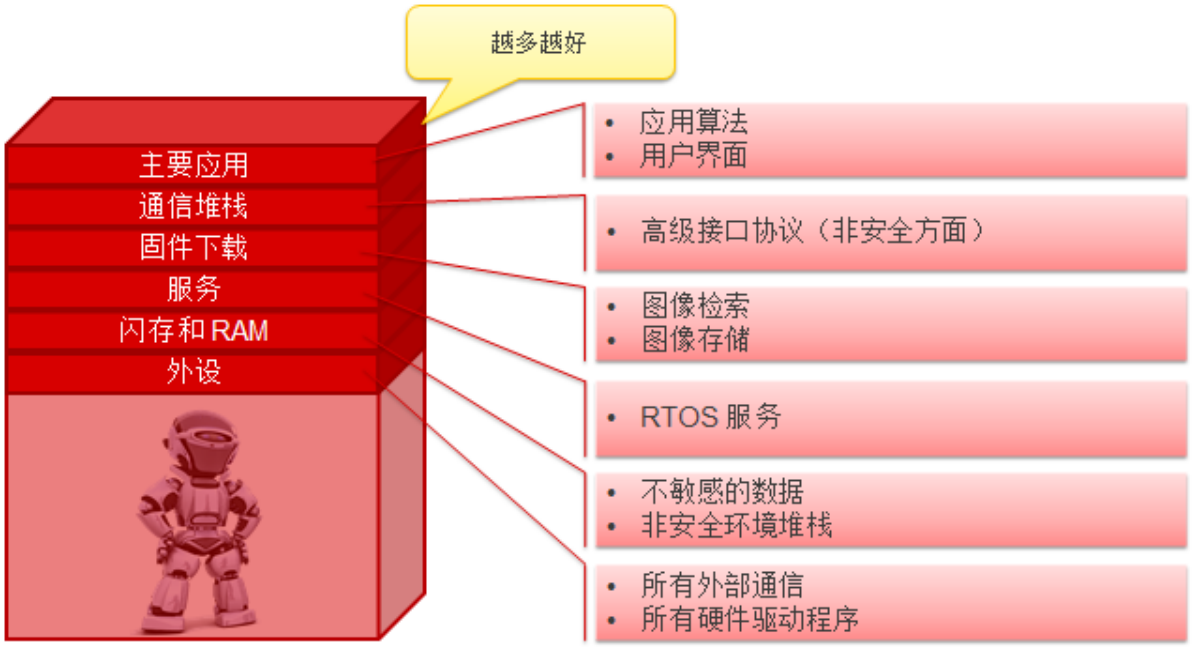


图 7：使用 Arm TrustZone 提供信任根保护

瑞萨 RA 产品家族中的 Cortex-M33 微控制器

瑞萨 RA 产品家族中 基于 Arm Cortex-M33 的微控制器，采用 Arm v8-M TrustZone 技术。该微控制器提供从 120 MHz 到 200 MHz 的多个性能配置，集成了高级安全加密引擎和加密加速器，可以实现安全芯片的功能。安全功能还包括支持密钥管理、篡改检测和防止功耗分析功能。该微控制器提供用于应用代码的 1 MB 闪存区域，以及包含 ECC 功能 的 256 KB SRAM。外设功能包括电容式触摸感应控制器、以太网、全速 USB 2.0 、SDHI 和 SPI 接口。

RA 产品家族 的开发支持包括瑞萨灵活配置软件包 (FSP)，这是一个基于 GitHub 且易于使用的生态系统，在该生态系统上创建安全的物联网应用。FSP 包括用于连接、安全性、图形和硬件抽象层 (HAL) 驱动程序等功能的中间件。FSP 中还包含板极支持包和 FreeRTOS。

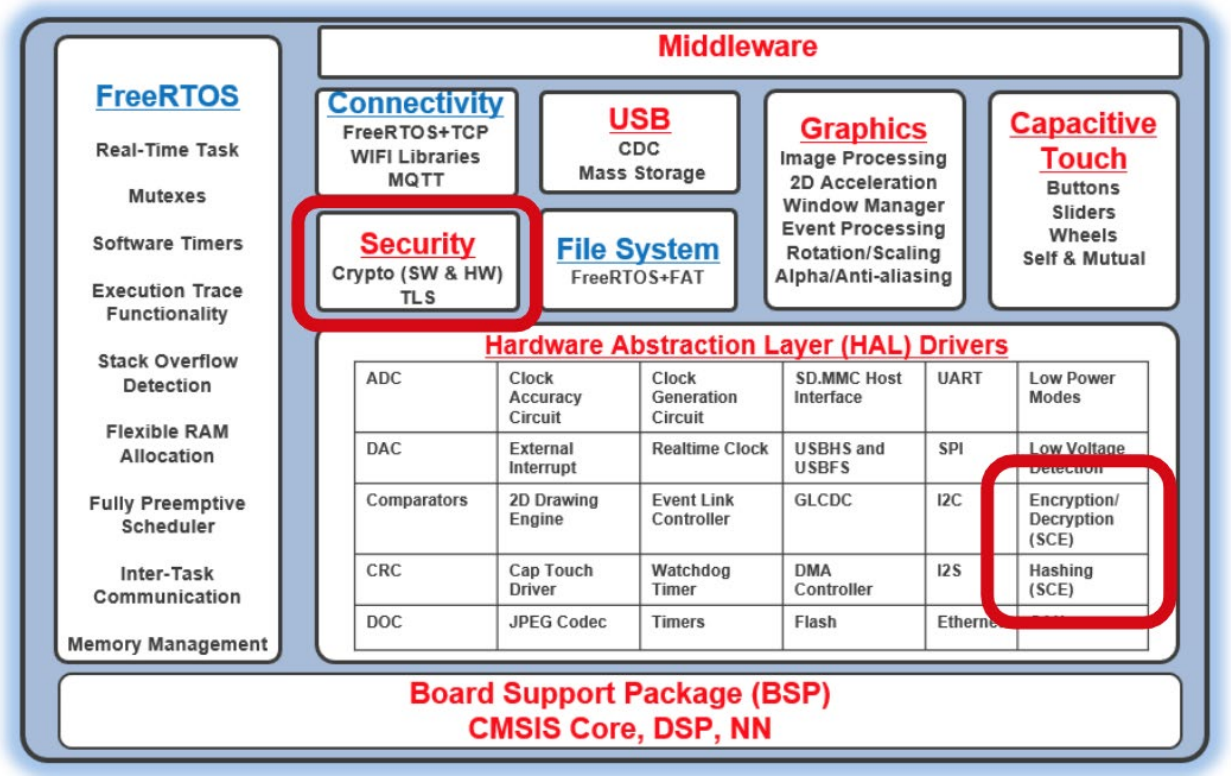


图 8：FSP 功能概述

瑞萨 RA 合作伙伴生态系统 列出了提供硬软件构建模块与资源的各个合作伙伴，这些构建模块和资源能够加速物联网应用的开发。

结论

如需构建需要逻辑分离和隔离代码、安全性与 IP 资产的各类工业与消费类应用，瑞萨 RA 产品家族中基于 Cortex-M33 的高性能、低功耗微控制器是理想的平台。瑞萨通过在 RA6M4 MCU 上实现 Arm TrustZone，增强了 TrustZone 的功能，并将外设接口、RAM 和闪存、直接内存访问请求都分离开来。

了解更多

1. RA6M4 产品页面
2. RA4M3 产品页面
3. RA 合作伙伴生态系统
4. 灵活配置软件包 (FSP)
5. RA MCU 系列

6. Arm® TrustZone®