AN12326

Secure GPIO 和使用方法

版本 1 — 2020 年 2 月 26 日 应用笔记

目录

1 背景

LPC55Sxx（带有 TrustZone 功能）具有 Secure GPIO 模块，其使用与普通 GPIO、TrustZone 和 Secure AHB Controller 密切相关。本节简要介绍这些功能。有关更多信息，请参阅用户手册（User Manual）。

1.1 TrustZone 和 Secure AHB Controller

1.1.1 TrustZone

Armv8-M 的 TrustZone 可用于保护安全资源免受恶意代码的侵害。安全资源包括安全存储器（代码/数据）和安全外围设备。它是通过将地址空间分割成安全(S)或非安全(NS)来实现的。TrustZone 可以根据分配给该地址空间的特定安全属性(S，NS)过滤来自 CPU0 的地址访问。如 图 1 所示，安全状态下的 CM33 CPU(CPU-S)可以执行来自安全存储器(S-memory)的指令，但不允许直接从非安全存储器(NS-memory)执行指令。然而，CPU-S 可以访问 S-内存和 NS-内存中的数据。CPU-NS 只能从 NS-memory 执行指令，不允许从 S-memory 执行指令。CPU-NS 只能在 NS-memory 中访问数据，不允许从 S-memory 访问数据。

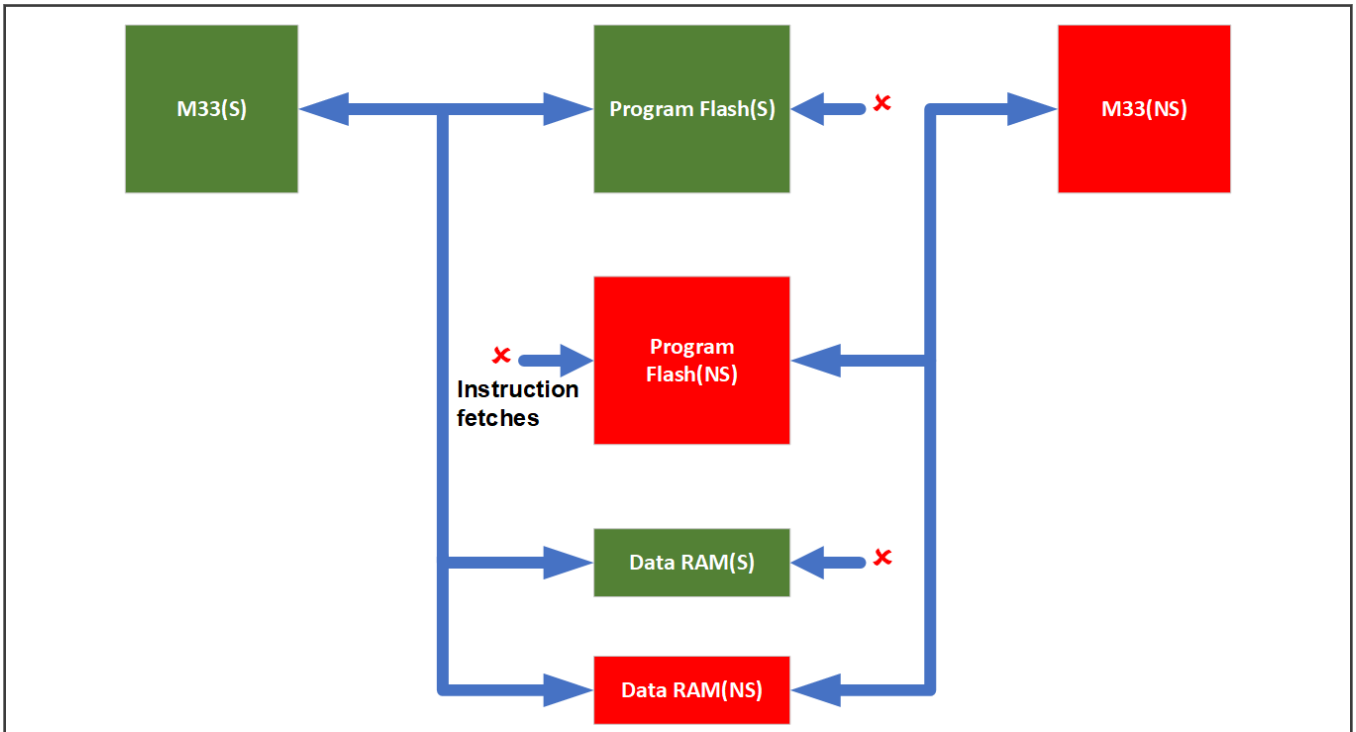

图 1. 信任区的安全状态和非安全状态视图

综上：

- 非安全(NS)的应用程序“信任”安全的代码(secure code)不会无意中损坏/修改 NS 代码或数据，或故意造成故障或危险。
- 安全(S)的应用程序代码不“信任”非安全(NS)的应用程序代码，不允许 CPU-NS 的访问。

1.1.2 Secure AHB Controller

LPC55Sxx（带有 TrustZone）使用 Secure AHB Controller 实现第二层保护，以在系统级别上提供安全执行。

使用 Secure AHB Controller，可以为每个外围设备配置安全访问规则。

默认情况下，处于安全状态(CPU-S)的 CM33CPU 可以访问 S 态和 NS 态的外围设备。不安全状态下的 CM33CPU（CPU-NS）只能访问 NS 状态下的外围设备。如 图 2 所示。

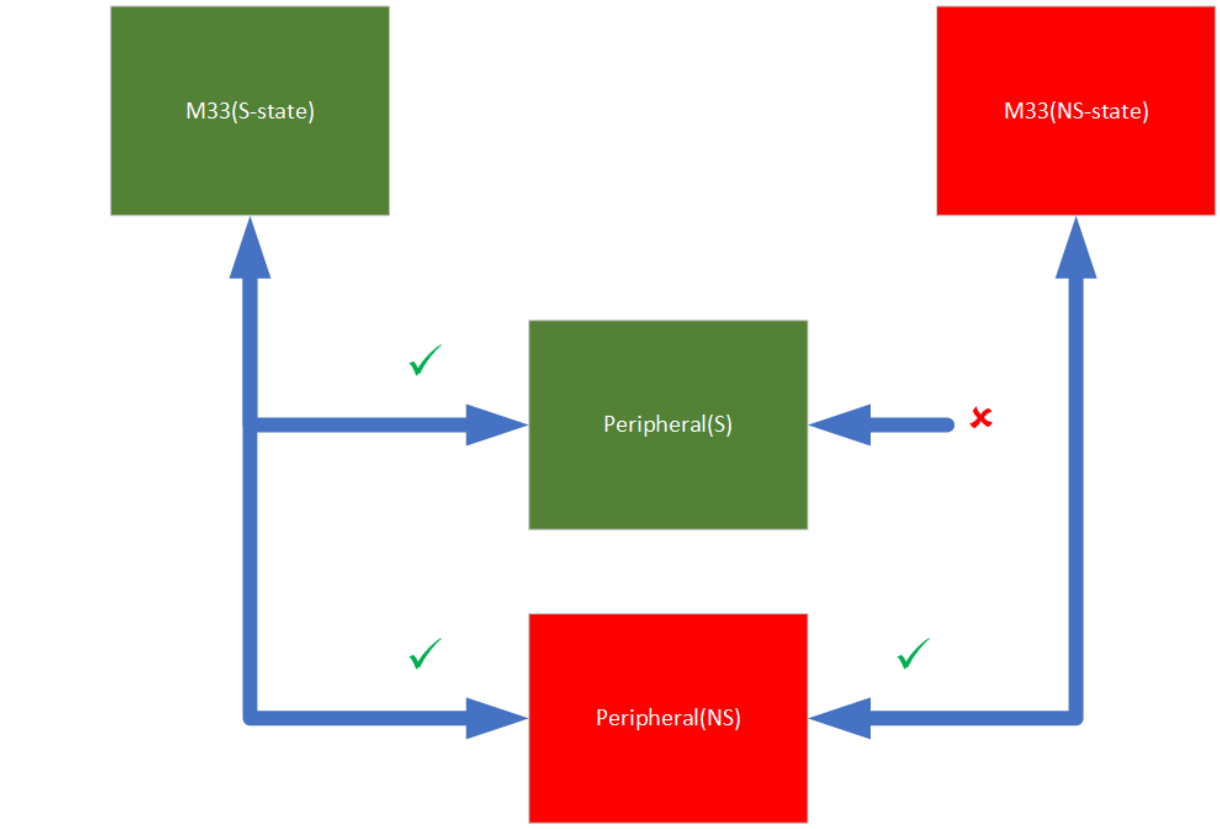

图 2. 信任区和安全 AHB 控制器的安全状态和非安全状态

1.2 普通 GPIO

普通 GPIO 是微控制器中最常见的数字外设。LPC 单片机的普通 GPIO 是非常灵活和强大的。像 SPI、UART 等，普通 GPIO 也是单片机中的数字外设。下面是普通 GPIO 的简单框图。不管引脚的功能被如何配置，普通 GPIO 都可以读取引脚的状态。例如，如果这个引脚被配置为 UART，那么引脚状态也可以通过普通 GPIO 读取。

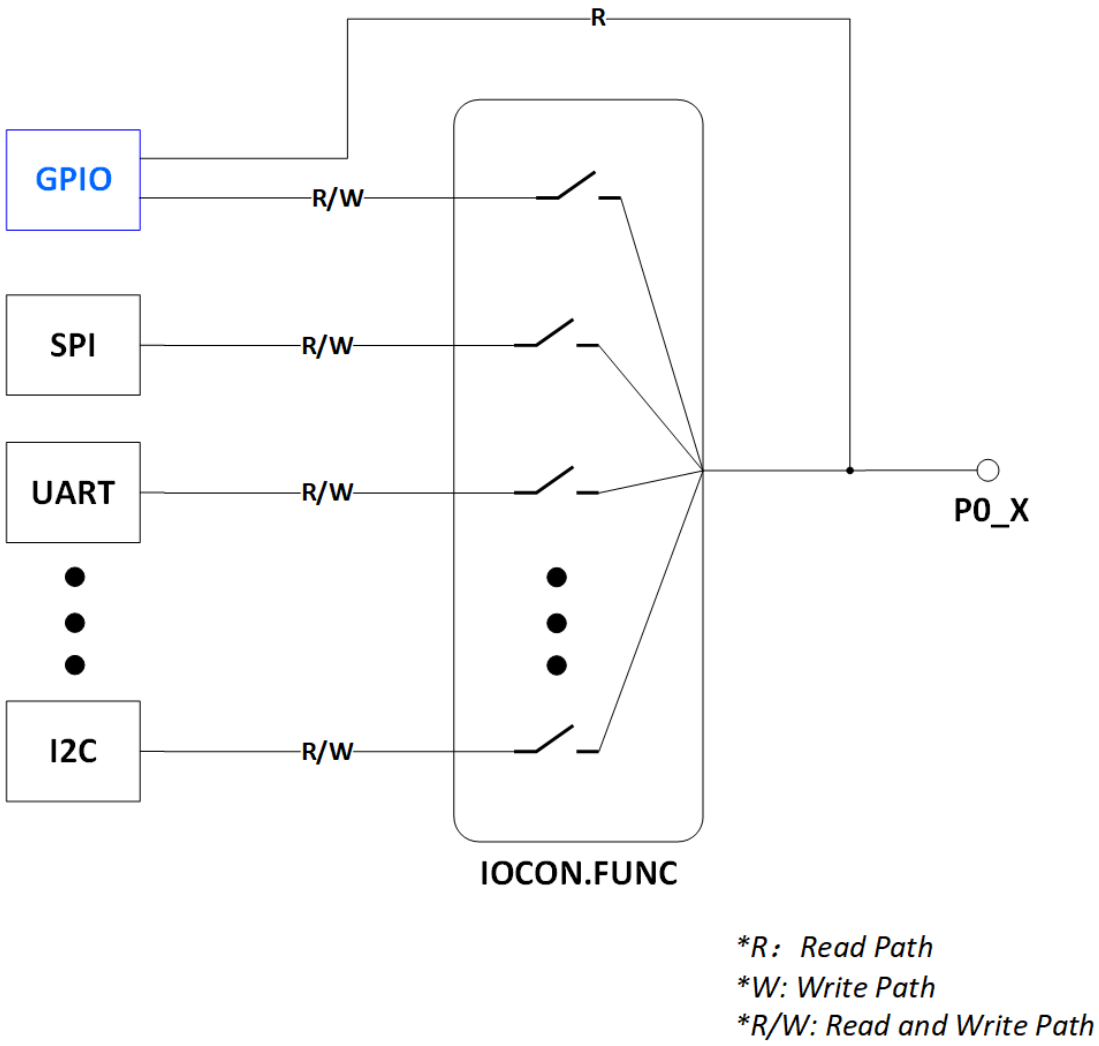

图 3. 普通 GPIO

2 Secure GPIO，Secure GPIO Mask 和 Secure PINT

由于普通 GPIO 的架构，所有数字 IO 引脚状态都是通过普通 GPIO 模块从 GPIO 读取路径读取的，与此引脚选择的功能无关。因此，有可能从安全资源(S)泄漏信息。例如，当 UART 被配置为安全外设时，这意味着这个 UART 只允许被安全世界访问(即安全代码)，而不能由非安全世界访问。然而，在这种情况下， UART 引脚状态仍然可以通过普通的 GPIO 读取路径被非安全世界监视，如 图 3 所示。因此，非安全世界可以获得所有安全 UART 的信息。

为了解决这个问题并保护来自安全外设上的数据，在 LPC55Sxx（带有 TrustZone）上实现了 Secure GPIO Mask。此外，如果安全世界需要操作 GPIO，则不能使用普通的 GPIO，因为普通的 GPIO 被屏蔽。在这种情况下，在 LPC55Sxx（带有 TrustZone）上引入了一个名为 Secure GPIO 的新模块。与普通 GPIO 不同，只有当 FUNC 在 IOCON 中=10 时，这个 Secure GPIO 功能才可用。

出于同样的原因，Secure-world 需要 Secure Pin Interrupt/Pattern Match Engine(PINT)，因此实现了另一个 Secure PINT 的模块。图 4 是 Secure GPIO 和 Secure GPIO Mask 的简单框图。

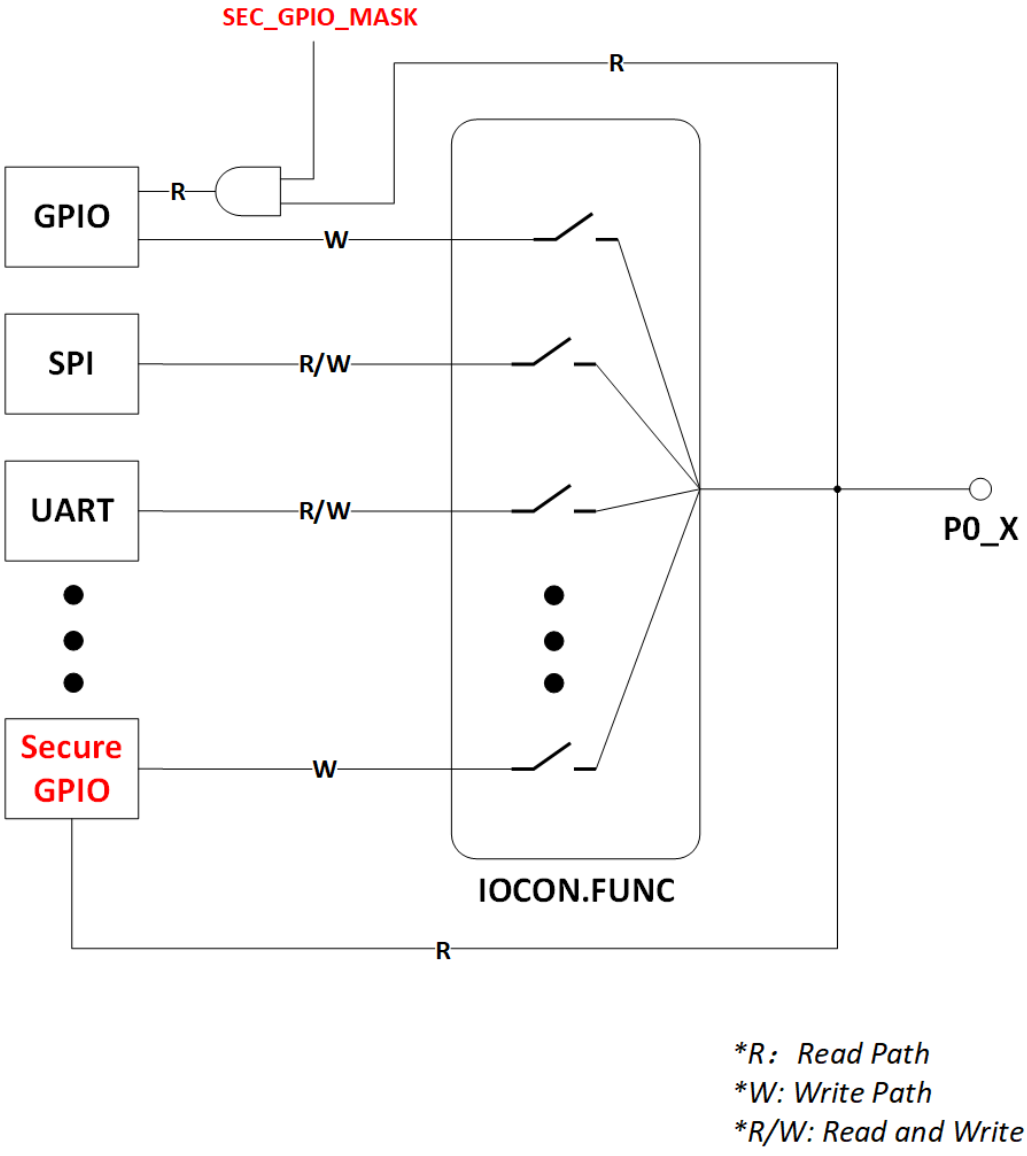

图 4. 安全 GPIO 和安全 GPIO 面罩

2.1 Secure GPIO Mask

每个 GPIO 都有一个 Secure GPIO Mask。如 图 4 所示，我们可以认为 Secure GPIO Mask 是 AND 门的一个输入。其默认值为 1。通过安全 GPIO 掩码，可以控制普通 GPIO 读取路径的开/关状态。

2.2 安全的 GPIO

如 图 4 所示，Secure GPIO 具有与普通 GPIO 相同的功能。然而，对于不同安全级别的 Secure GPIO 访问规则通过 Secure AHB Controller 配置，Secure AHB Controller 只能在安全状态下访问。

2.3 Secure PINT

Secure PINT 和 PINT 的主要区别是，Secrue PINT 只支持端口 0 上的最多两个引脚。与 Secure GPIO 类似，对该模块的访问规则是通过 Secure AHB Controller 配置的。安全引脚中断生成器和安全模式匹配引擎可在所有 LPC55Sxx（带有 TruztZone）设备上使用。与普通 PINT 类似，安全引脚中断生成器和安全模式匹配引擎是互斥的。

2.3.1 安全引脚中断

- 对于 Secure PINT 模块，最多可以从端口 0 上的所有引脚中选择两个引脚，作为边沿触发或电平触发的中断请求。每个请求在 NVIC 中创建一个单独的中断。
- 边沿触发中断引脚可以在上升沿或下降沿产生中断或都产生中断。
- 电平触发中断引脚可以是高电平触发或低电平触发。

2.3.2 安全模式匹配引擎

- 最多可以从端口 0 上的所有数字引脚中选择两个引脚来组成布尔表达式。布尔表达式由这些引脚的各种组合上的指定级别与/或转换组成。
- 包含指定布尔表达式的每个位片最小项（生成项）都可以生成自己的专用中断请求。
- 模式匹配事件的发生可以生成一个 RXEV 通知到 CPU。
- 模式匹配可以与软件结合使用，以创建基于引脚输入的复杂状态机。

3 用法

3.1 使用 Secure GPIO Mask 来保护需要使用 IO 的安全数字外设

SEC_GPIO_MASK 寄存器用于控制 Secure GPIO Mask。默认寄存器值都是 1，这意味着 NS 代码仍然可以通过读取其引脚状态来读取安全外设状态，如 图 5 左侧所示。

为了防止这种安全信息泄漏的风险，普通 GPIO 应通过将 SEC_GPIO_MASK 中的相应位设置为 0 来屏蔽，如 图 5 右侧所示。

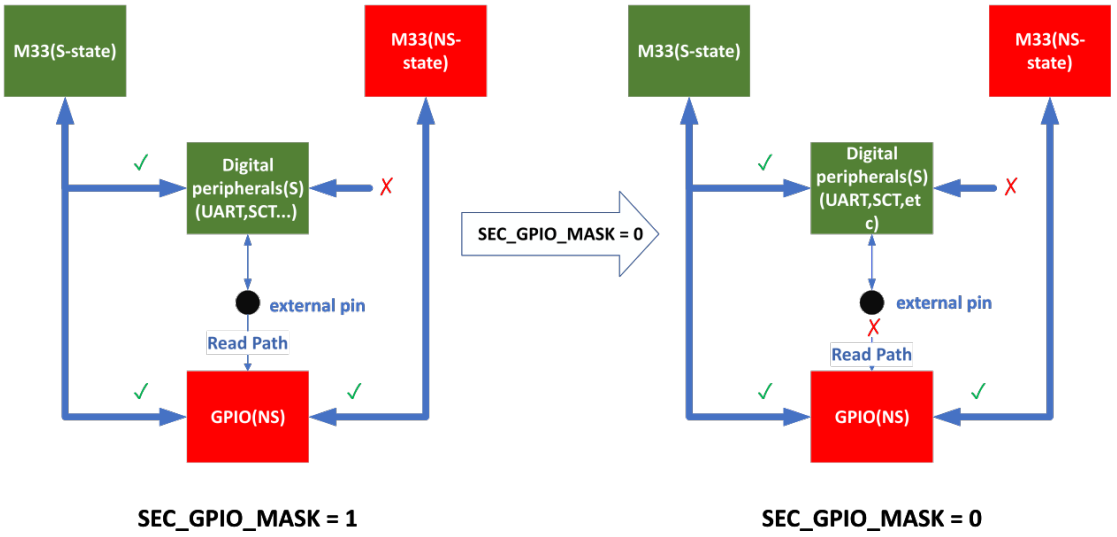

图 5. SEC_GPIO_MASK 的用法

下面的代码片段展示了如何使用 Secure GPIO Mask 屏蔽 P0_5 引脚：

```
AHB_SECURE_CTRL->SEC_GPIO_MASK0 = AHB_SECURE_CTRL->SEC_GPIO_MASK0 &
                                  ~AHB_SECURE_CTRL_SEC_GPIO_MASK0_PIO0_PIN5_SEC_MASK(0x1U);
```

图 6. 将 P0_5 的 SEC_GPIO_MASK 设置为 0

3.2 把一个 IO 设置为 Secure GPIO

以下是将一个 I/O 引脚配置为 Secure GPIO 的步骤：

1. 将相应的 SEC_GPIO_MASK 位配置为 0。
2. 配置 Secure GPIO 模块，通过 Secure AHB Controller 进行安全保护，防止非安全世界访问 Secure GPIO。
3. 通过 Secure AHB Controller 将 IOCON 模块配置为安全，防止非安全世界访问 IOCON。
4. 配置相应的引脚功能，通过 Secure IOCON 块将引脚功能配置为 Secure GPIO(FUNC=10)。
5. 启用 Secure GPIO 的时钟。

然后，你可以像普通 GPIO 引脚一样使用它。

下面的代码片段以 P0_5 引脚为例。

- 将 P0_5 的 SEC_GPIO_MASK 配置为 0：

```
AHB_SECURE_CTRL->SEC_GPIO_MASK0 = AHB_SECURE_CTRL->SEC_GPIO_MASK0 &
                                  ~AHB_SECURE_CTRL_SEC_GPIO_MASK0_PIO0_PIN5_SEC_MASK(0x1U);
```

图 7. 将 P0_5 的 SEC_GPIO_MASK 配置为 0

- 将 Secure GPIO 设为安全：

```
AHB_SECURE_CTRL->SEC_CTRL_AHB2[0].SEC_CTRL_AHB2_0_SLAVE_RULE = (uint32_t)(0x3U);
```

图 8. 将 Secure GPIO 设为安全

```
AHB_SECURE_CTRL->SEC_CTRL_APB_BRIDGE[0].SEC_CTRL_APB_BRIDGE0_MEM_CTRL0 =
        AHB_SECURE_CTRL_SEC_CTRL_APB_BRIDGE_SEC_CTRL_APB_BRIDGE0_MEM_CTRL0_IOCON_RULE(0x3U);
```

图 9. 使 IOCON 块安全

- 配置 P0_5 引脚功能以保护 GPIO (FUNC=10)

```
const uint32_t port0_pin5_config = (/* Pin is configured as Secure GPIO */
                                    IOCON_PIO_FUNC10 |
                                    /* No addition pin function */
                                    IOCON_PIO_MODE_INACT |
                                    /* Input function is not inverted */
                                    IOCON_PIO_INV_DI |
                                    /* Enables digital function */
                                    IOCON_PIO_DIGITAL_EN |
                                    /* Standard mode, output slew rate control is enabled */
                                    IOCON_PIO_SLEW_STANDARD |
                                    /* Open drain is disabled */
                                    IOCON_PIO_OPENDRAIN_DI);
/* PORT0 PIN30 (coords: A2) is configured as FC0_TXD_SCL_MISO */
IOCON_PinMuxSet(IOCON, 0U, 5U, port0_pin5_config);
```

图 10. 配置 P0_5 引脚功能以安全 GPIO (FUNC=0xA)

- 启用安全 GPIO 时钟

```
CLOCK_EnableClock(kCLOCK_Gpio_Sec);
```

图 11. 启用安全 GPIO 时钟

3.3 Secure PINT 的使用

从应用的角度来看，Secure PINT 的使用方法与普通 PINT 相同。有一点需要特别注意：

- 若要禁用非安全世界访问 Secure PINT 寄存器，请通过 Secure AHB Controller 将 Secure PINT 设置为安全。
- 然后，您可以像普通 PINT 一样使用它，并使用与普通 PINT 相同的 API。

上述设置的代码片段如下所示。

将 Secure PINT 设置为安全：

```
/* Set Secure PINT register as secure */
AHB_SECURE_CTRL->SEC_CTRL_APB_BRIDGE[0].SEC_CTRL_APB_BRIDGE0_MEM_CTRL0 =
    AHB_SECURE_CTRL_SEC_CTRL_APB_BRIDGE_SEC_CTRL_APB_BRIDGE0_MEM_CTRL0_SEC_PINT_RULE(0x3U);
```

图 12. 使安全 PIN 寄存器最安全

4 示例

本章以 LPC55S69 为例，其他设备的操作类似。

4.1 环境

4.1.1 硬件环境

- 开发板
  - LPCXpresso55S69
- 调试器
  - 板上集成 CMSIS-DAP 调试器
- 杂项
  - 一根 Micro USB 线
  - 个人电脑
- 板的设置
  - 使用 USB 线将 PC 与板子的 P6 相连

4.1.2 软件环境

- 工具链
  - IAR
- 软件包
  - AN_SecureGPIO_Demo.zip

4.2 步骤和结果

此示例演示如何使用 Secure GPIO。基本步骤如下：

1. 配置

   打开位于路径中的“secure_gpio_s”项目，如下所示。

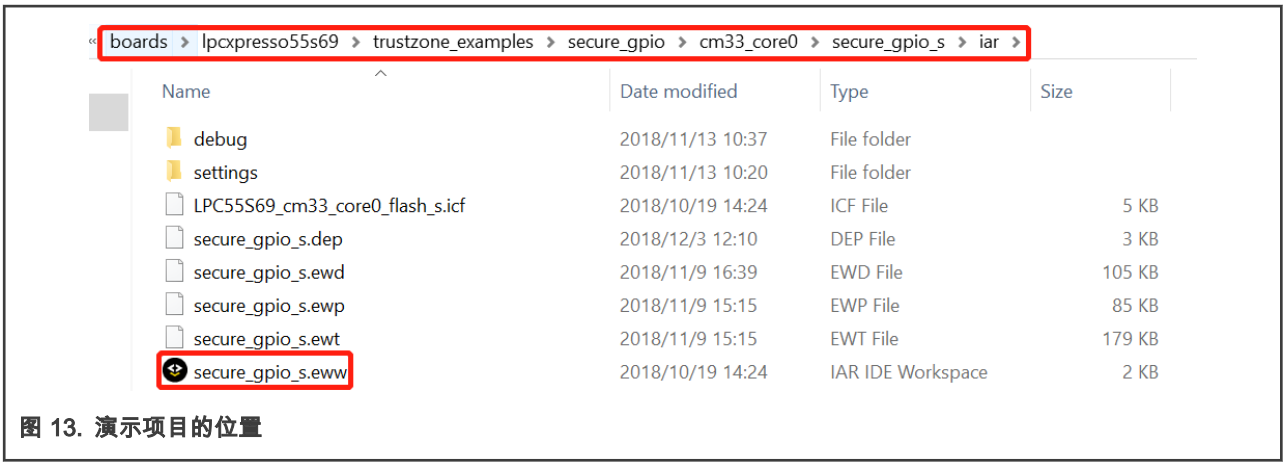

图 13. 演示项目的位置

工作区中有两个项目。

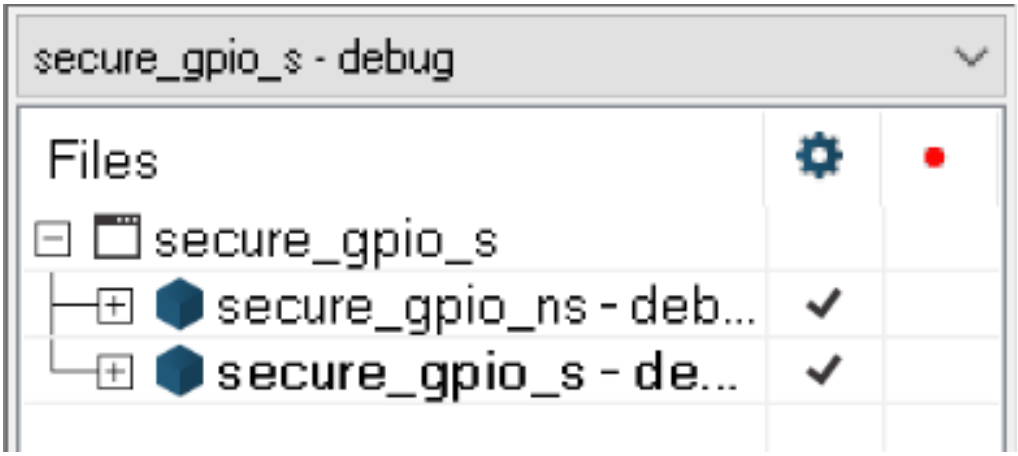

图 14. 演示项目

配置“secure_gpio_s”和“secure_gpio_ns”项目，如下所示。

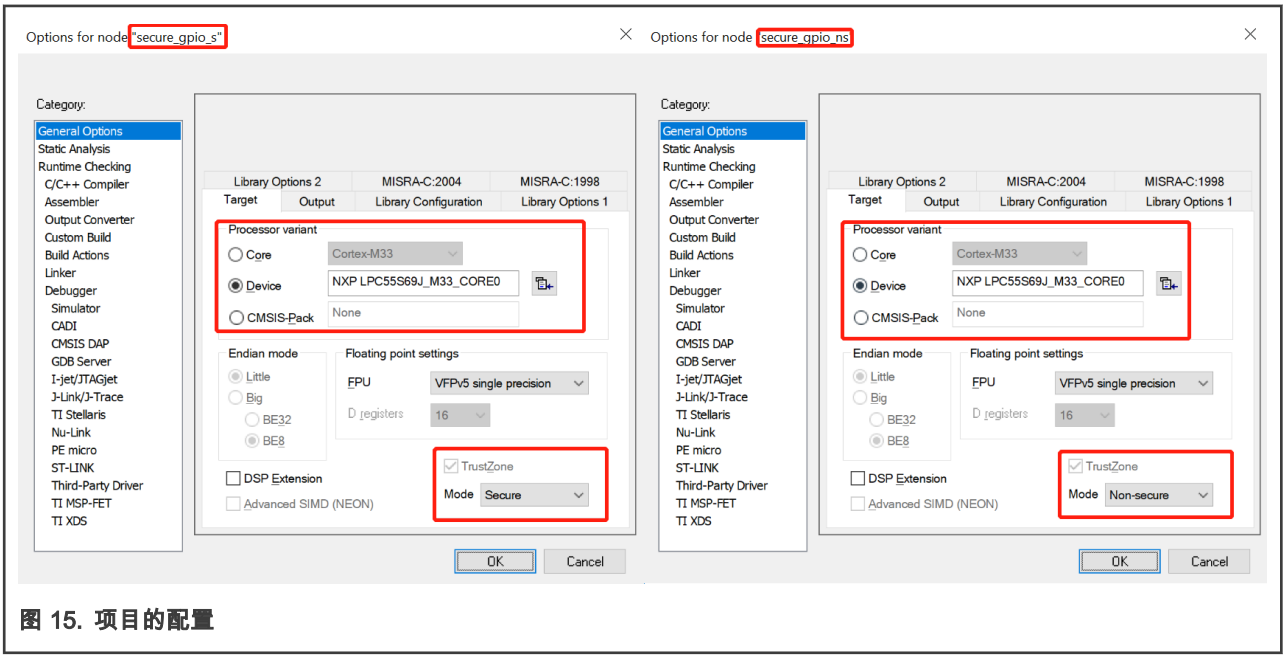

图 15. 项目的配置

2. 编译和下载
   - 先编译“secure_gpio_s”项目，然后编译“secure_gpio_ns”项目。
   - 使用 USB 线将 PC 与板子的 P6 相连，同时按住 ISP 按钮。
   - 下载编译后生成的可执行文件。
   - 下载成功后松开 ISP 按钮。
3. 运行

   通过按下板上的复位(S4)按钮来复位运行。
4. 结果

   在这个例子中使用了两个 LED。蓝色 LED 表示引脚状态由普通 GPIO 读取，而绿色 LED 表示引脚状态由 Secure GPIO 读取。复位后，代码在安全世界中运行，它初始化系统，包括上面的两个 LED，然后跳转到非安全世界。在非安全世界中，它通过普通 GPIO 和安全 GPIO 读取 P0_5 引脚(EVK 上的 ISP 按钮/S1)，并且它读取的引脚状态为 1，因为默认情况下，该引脚是外部上拉的。当 ISP 按钮被按下并按住时，P0_5 将被读取为 0，如果 P0_5 为 0，则打开蓝色 LED 和绿色 LED，因为现在普通 GPIO 和 Secure GPIO 都从这个引脚读取且都为 0。按 USER 按钮(S3)，它跳转到安全世界，切换 Secure GPIO Mask，然后跳转回非安全世界。按 WakEUP 按钮(S2)，它将跳转到安全世界，将 Secure GPIO 设置为安全，然后跳回到非安全世界。最后，它试图从非安全世界访问安全 GPIO，由于安全访问违规，它进入 Hard Fault。下图简单描绘了这个流程：

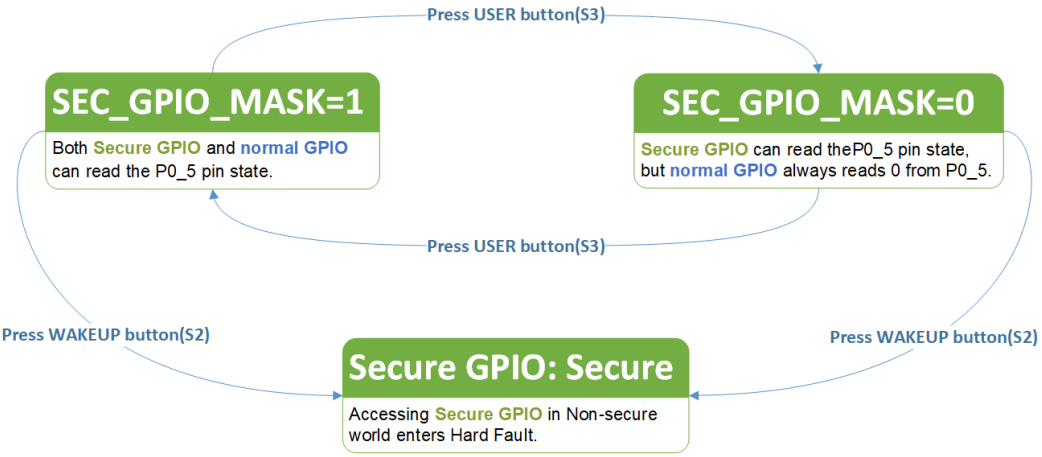

图 16. 安全 GPIO 示例流程

5 结论

这个例子表明，无论引脚功能和外设功能是安全的还是非安全的，非安全世界都可以访问外设引脚状态。这导致了安全信息泄漏。为了防止信息泄露，必须使用 Secure GPIO，并应在安全世界中配置和使用。普通的 GPIO 应用于非安全世界。同样的规则适用于 Secure PINT 和普通 PINT。

6 修订记录

版本号	日期	说明
0	2019 年 1 月 15 日	初始版本
1	2020 年 2 月 26 日	一般更新

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Limited warranty and liability — Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

Right to make changes - NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Security — Customer understands that all NXP products may be subject to unidentified or documented vulnerabilities. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer’s applications and products. Customer’s responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer’s applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately. Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP. NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: 2020 年 2 月 26 日

Document identifier: AN12326

arm