SPC1169 Security 使用指南

版本 A/1 – 2023 年 9 月

概述

为了降低软件盗窃的风险，许多芯片都有芯片内部的程序保护。SPC1169 的设计考虑到了程序保护的重要性，为用户提供了强大可靠的程序保护功能，其中包括调试锁定和随机数保护。

目录

图片列表

表格列表

版本历史

版本	日期	作者	状态	变更
A/0	2023-4-11	CanChai	Outdated	首次发布。
A/1	2023-9-4	HangSu	Released	更新章节 1

术语或缩写

术语或缩写	描述

1 锁定芯片调试接口

当调试或烧录程序时，芯片的调试接口经常被使用。这也意味着通过调试接口可以读取或修改芯片内部存储器的内容。因此，必须在批量生产过程中禁用芯片的调试功能。否则，通过调试接口可以获取芯片内部程序。

通过设置配置字的 CHIP_SECURITY 字段，可以锁定 SPC1169 的调试接口。配置字的描述如表 1-1 所示。一旦 SPC1169 的调试接口被锁定，无法通过调试接口访问内部存储器。此外，通过 ROM 中的引导加载程序读取、编程和扇区擦除内部存储器也被禁用；ROM 中的引导加载程序仅支持擦除 Flash 存储器。

通过以下方法可以解锁 SPC1169 的调试接口：

- 通过 ROM 中的引导加载程序对整个内部 Flash 存储器进行芯片擦除。
- 通过 ROM 中的引导加载程序输入正确的非安全密钥（等于配置字的 UNSECURITY_KEY 字段）。此方法临时解锁调试接口。这意味着芯片重新启动时调试接口将保持锁定状态。如果 UNSECURITY_KEY 字段的值为 0x00000000_00000000，则表示无法通过输入非安全密钥解锁调试接口。

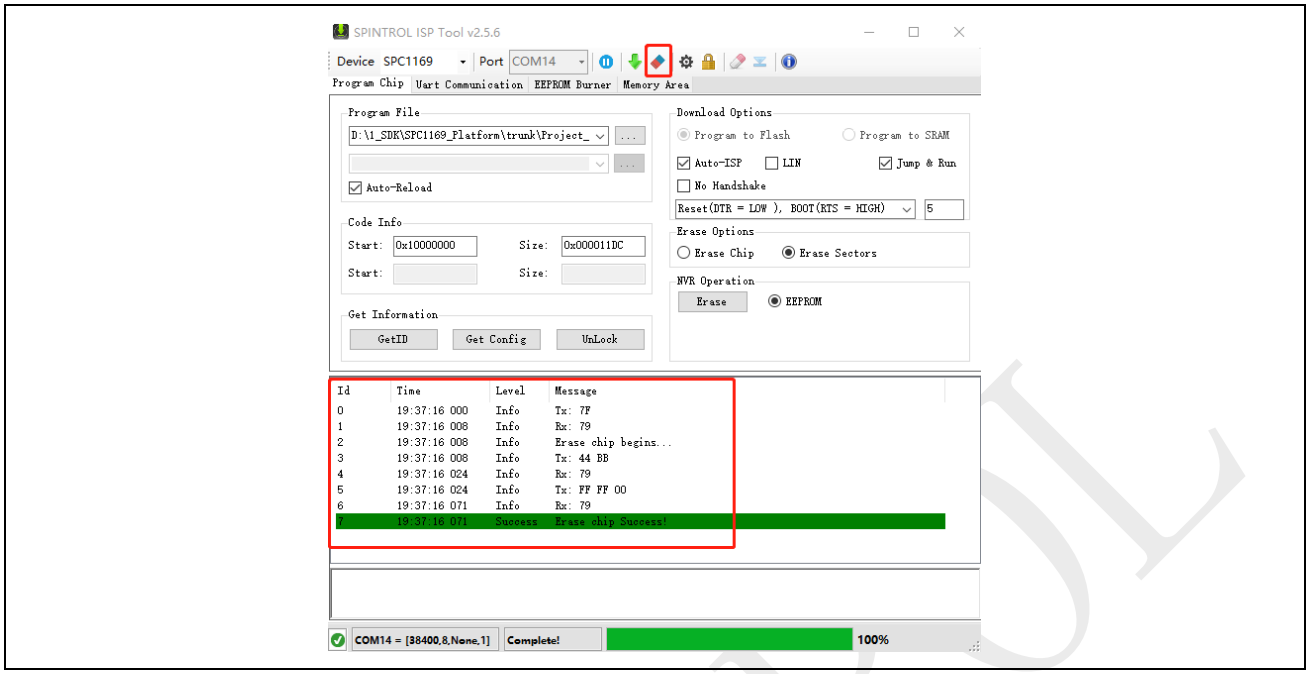
表 1-1：配置字的描述

地址	名字	描述
0x1001FFF0	UNSECURITY_KEY	解锁芯片的调试接口需要输入一个 8 字节的密钥 UNSECURITY_KEY，当调试接口被锁定时，只有正确的密钥才能使调试接口暂时解锁。如果 UNSECURITY_KEY 字段的值全部为 0x00，则不能通过输入密钥来解锁调试接口。
0x1001FFF8	WDT_ENABLE	Watchdog 使能字 0xFFFFFFFF：在芯片启动时禁用看门狗 其他：在芯片启动时启用看门狗
0x1001FFFC	CHIP_SECURITY	芯片调试接口锁定字 0xFFFFFFFF：芯片调试接口将不被锁定 其他值：芯片调试接口将被锁定

下面结合 SPINTROL ISP Tool v2.5.6，演示了 CHIP_SECURITY 功能。

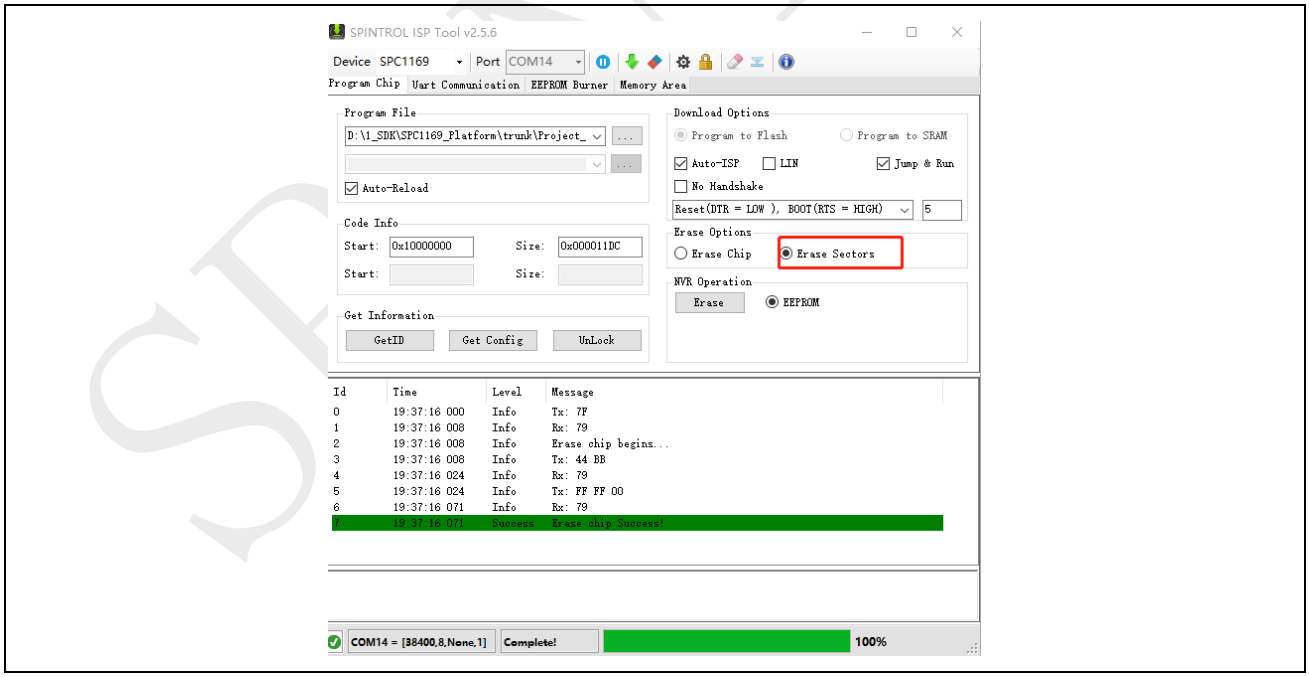
- 按下复位键，之后点击 SPINTROL ISP Tool 中的 main flash 擦除按键，对芯片 main flash 进行擦除，确保 CHIP_SECURITY 为 0xFFFFFFFF，使芯片接口处于不被锁定的状态，如图 1-1 所示。

图 1-1：擦除 main flash



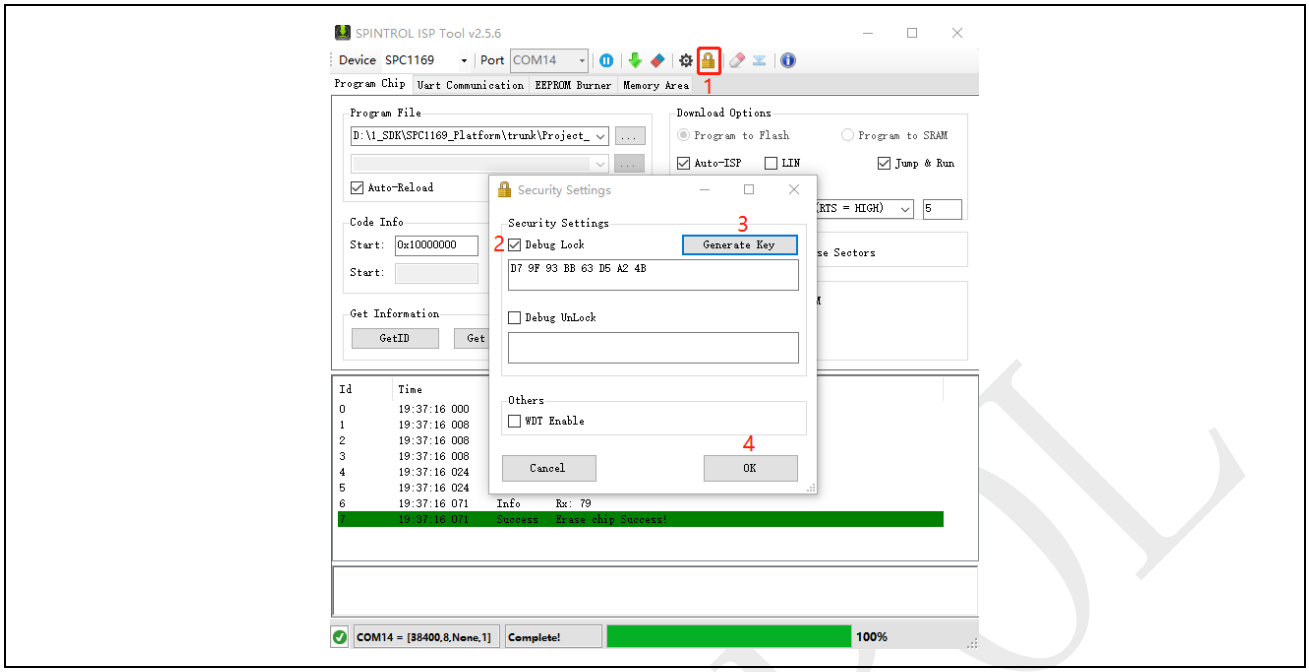
- 确保调试过程中一直选中 Erase sector，而不是 Erase chip，从而在接下来的调试过程中位于 main flash 中的 CHIP_SECURITY 不会被误擦除，如图 1-2 所示。

图 1-2：确保 Erase Sector 选中



- 使能 Debug Lock，并 Generate Key，如图 1-3 所示。

图 1-3：产生密码



- 按下复位键，随便下载一个代码，UNSECURITY_KEY, WDT_ENABLE, CHIP_SECURITY 会在下载的最后写入到 0x1001FFF0 开始的位置，如图 1-4 所示。

图 1-4：通过上位机下载密码到 main flash

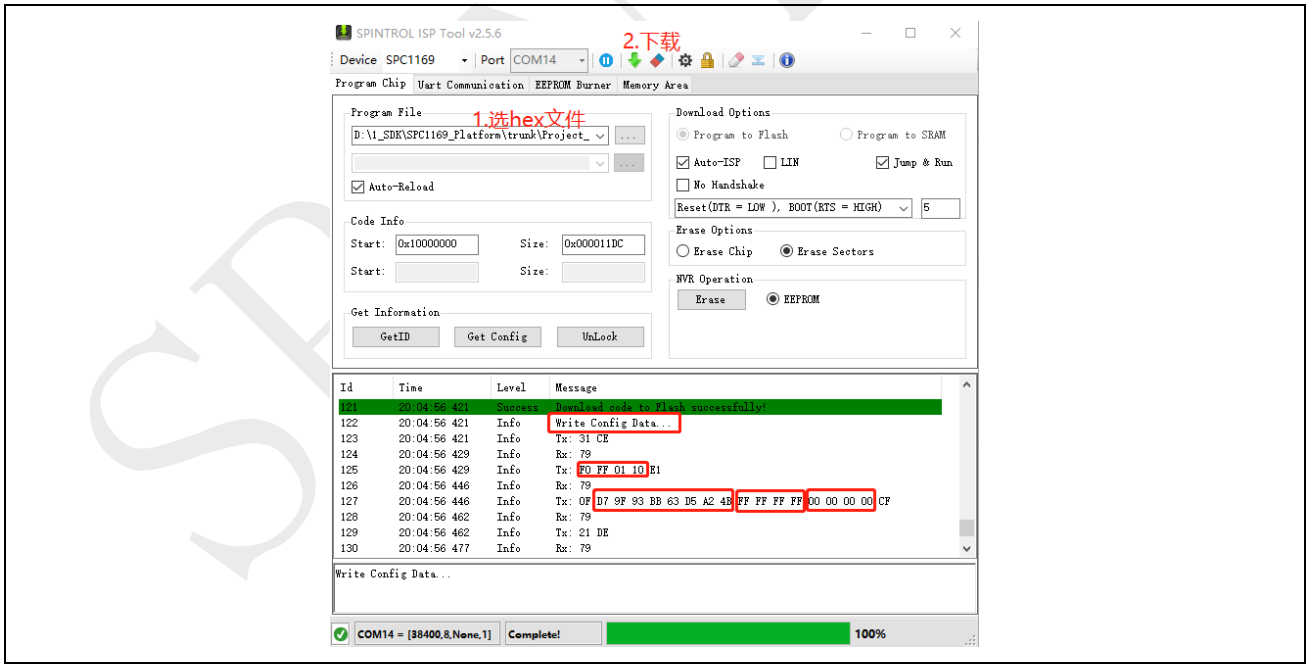
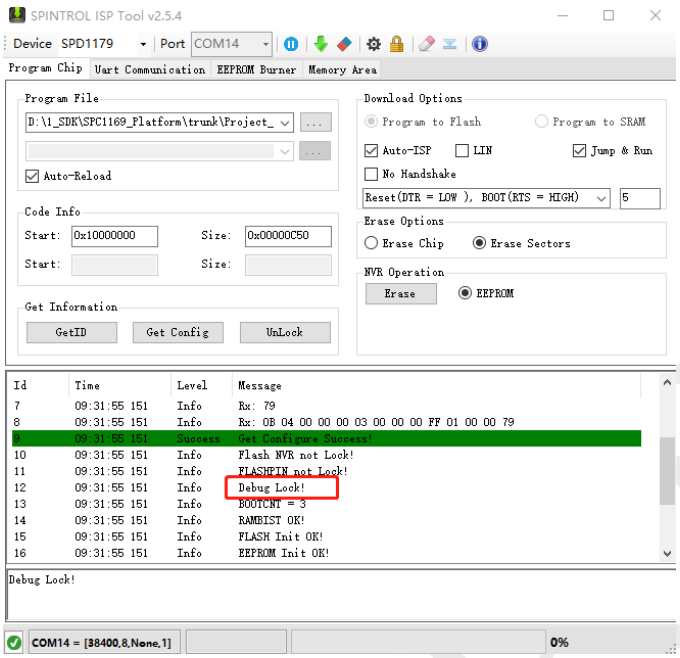


图 1-4 中 0xD7 0x9F 0x93 0xBB 0x63 0xD5 0xA2 0x4B 为 key，0xFF 0xFF 0xFF 0xFF 为 WDT_ENABLE，0x00 0x00 0x00 0x00 为 CHIP_SECURITY，如表 1-1 所示，芯片启动时的 WDT 被禁止，芯片调试接口将被锁定。

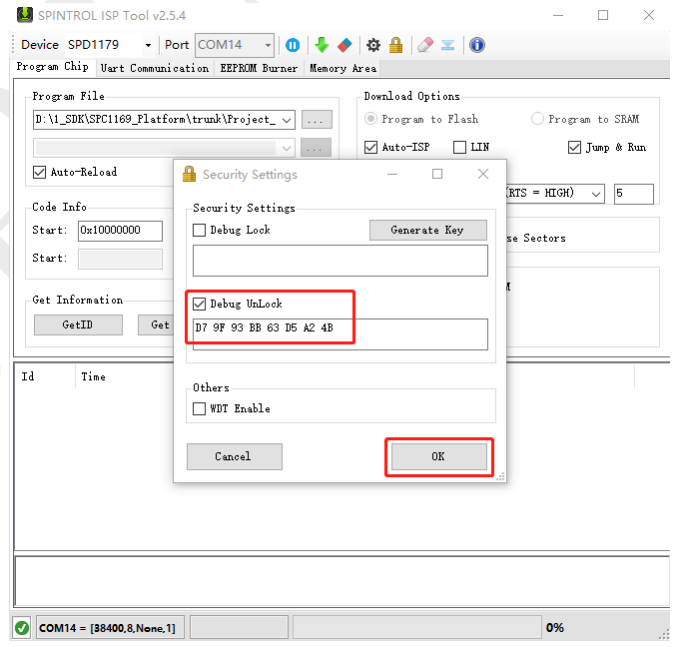
- 按下复位键，尝试下载代码，会发现握手失败，表明芯片调试接口已经关闭了。
- 平常也可以在按下复位键后点击 Get Config 查看芯片有无加锁，如图 1-5 所示。

图 1-5：Get Config



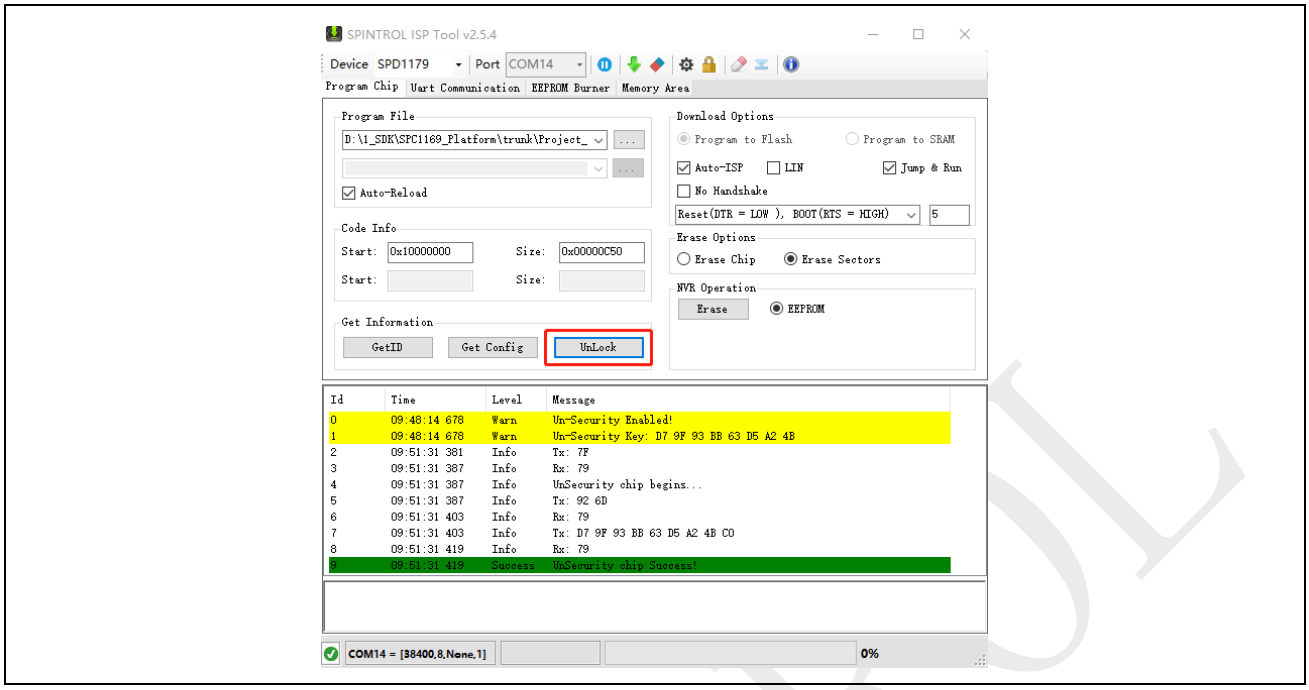
- 共有两种解锁方式，可以按下复位键，将控制字全部擦除，从而恢复 UNSECURITY_KEY, WDT_ENABLE, CHIP_SECURITY 到全 F 默认值，如图 1-1 所示。
- 也可以输入密码进行解锁，取消选中 Debug Lock，选中 Debug Unlock，并在 Debug Unlock 下输入之前生成的密码，如图 1-6 所示。

图 1-6：取消 Debug Unlock，并输入解锁密码



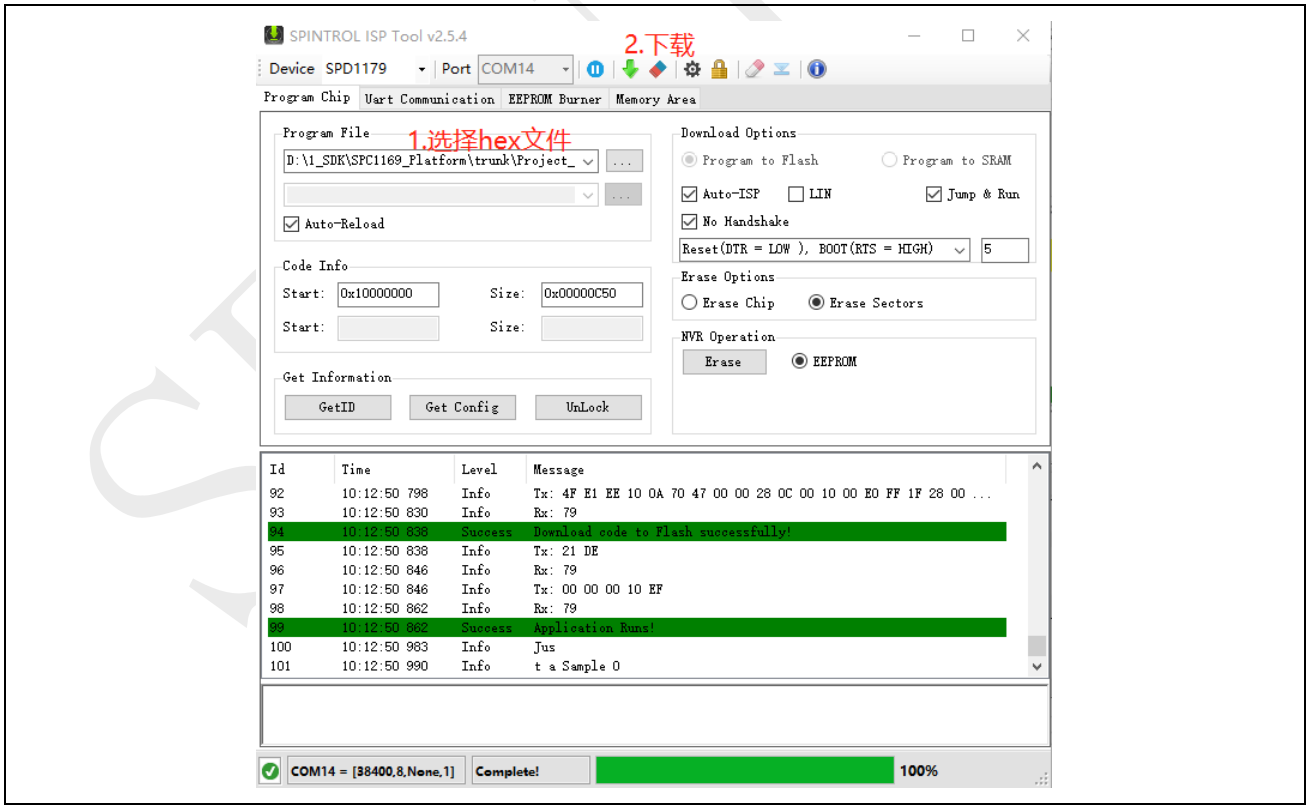
- 复位芯片，点击 Unlock 按钮，如图 1-7 所示，使用密码进行解锁为临时行为，Unlock 成功后，复位芯片解锁失效。

图 1-7：解锁



– 选中 No Handshake，随便下载一个代码，下载成功。

图 1-8：下载代码



注意：

正常使用时，需要取消勾选 No Handshake，否则芯片复位后，上位机与芯片无法通信。

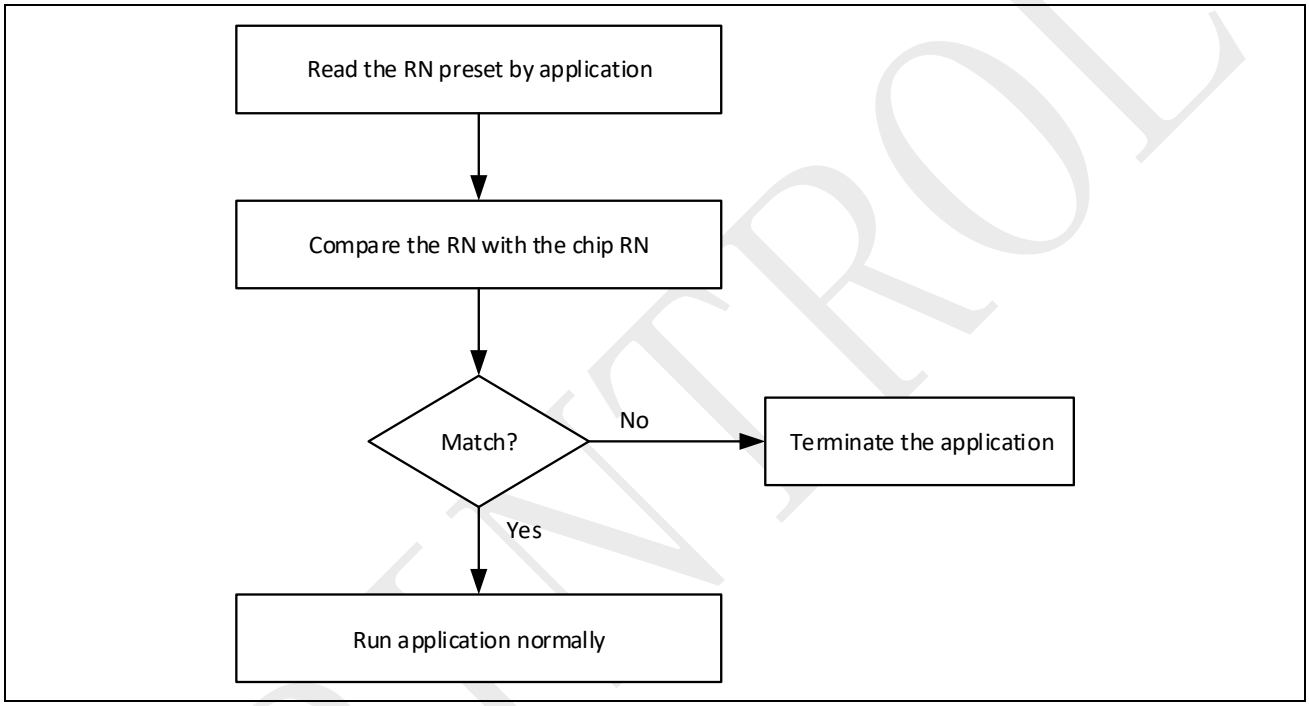
使用密码解锁只能在每次解锁成功后临时解锁下载接口，芯片复位后失效，要想永久解锁，可以将控制字全部擦除，从而恢复 **UNSECURITY_KEY, WDT_ENABLE, CHIP_SECURITY** 到全 **F** 默认值，如图 1-1 所示。

2 随机数据保护

有些芯片破解者可能有能力使用特定设备去除芯片封装，读取 Flash 内存中的数据，然后将其写入未编程的芯片以复制用户的产品。SPC1169 也针对这种情况具有适当的保护机制。

SPC1169 出厂时被写入一个 8 字节的随机数，该随机数无法修改。客户的应用程序可以在运行时从芯片中读取并验证随机数。如果随机数与应用程序预设的值不匹配，则应用程序将终止。由于芯片的随机数彼此不同，即使芯片破解者从一颗芯片的 Flash 存储器中获取程序数据，然后将程序数据写入其他 SPC1169 芯片，程序也无法正常工作。通过这种方式，客户的产品无法大量复制。

图 2-1：随机数在应用程序中的流程图



在图 2-1 中，应用程序预设的随机数（RN）可以在产品生产过程中通过编程器工具实现。编程器工具读取目标芯片的随机数，并将其写入目标芯片 Flash 存储器的特定地址。当应用程序启动时，写一段程序从目标芯片 Flash 存储器的特定地址读取预设的随机数，并将其与目标芯片的随机数进行比较。

3 安全功能的具体实现

对于在现场大规模生产中实现芯片安全性的步骤如图 3-1 所示。

图 3-1：实际生产过程中安全功能的具体实现流程图