STM32L5 系列微控制器和 TrustZone® 开发入门

引言

本文档为使用 EWARM 和 MDKARM 软件工具链在 STM32L5 系列微控制器上进行应用开发提供参考。

本应用笔记提供了为 Arm® Cortex®- m33（Armv8_M 架构）的器件构建和调试安全和非安全应用程序的基础知识。

本文首先概述 Arm® Cortex®-M33 和 TrustZone®概念。

本应用笔记还描述当通过 TZEN 选项位启用 TrustZone®之后，如何使用 EWARM 和 MDKARM 调试 STM32L5 系列微控制器。

1 概述

本文档适用于 STM32L5 系列单核 Arm®的微控制器。

提示 *Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.*

2 Arm® Cortex®-M33 内核概述

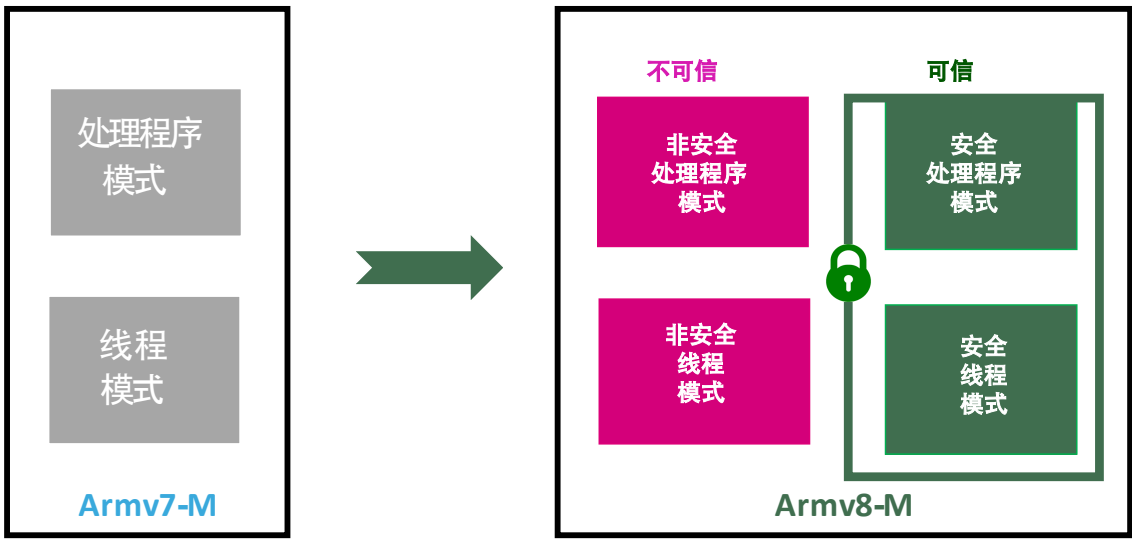
Arm® Cortex®-M33 是首次完整的集成了 ARMv8 指令集及 TrustZone 安全技术和数字信号处理功能。该处理器支持大量灵活的配置选项，以促进各种应用程序的部署，并提供专用的协处理器接口，用于加速经常使用的计算密集型操作。Cortex®-M33 提供性能、功率、安全性和生产效率之间的最佳平衡。

3 Armv8-M 的 TrustZone®概念

带有 TrustZone®的 Cortex®-M33 处理器有两个安全状态（参见图 1）和一些相关的特性：

- 安全状态
- 非安全状态
- 四个堆栈和四个堆栈指针寄存器
- 硬件栈限制检查
- 支持类似于可编程 MPU 的安全属性单元（SAU）
- 系统安全通知接口
- 限定非安全（NS）域只能通过预定义的入口点访问安全代码
- 当切换到非安全时，异常硬件自动保存和清除安全寄存器状态
- 中断或异常控制的扩展存储，SysTick
- 针对每个安全和非安全部分的内存保护单元。

图 1. Armv8-M 中的安全状态



提示 *TrustZone®被启用后，系统默认在安全状态下启动。*

4 SAU / IDAU - TrustZone®概念

TrustZone® 安全功能由 FLASH_OPTR 寄存器中的 TZEN 位激活。TrustZone®被启用后，SAU 和 IDAU 根据安全和非安全状态定义访问权限。

- IDAU：将第一个内存分区配置为安全或非安全可调用属性。IDAU 内存映射分区是不可配置的，它由硬件配置决定。
- SAU：8 个区域，用于覆盖 IDAU 以设置安全区域和确认非安全区域。
- 首先根据 IDAU 安全属性选择安全状态，然后结合 SAU 安全属性选择安全状态。最终的安全属性是 IDAU 和 SAU 的最高安全设置。
- “安全”安全属性具有最高的安全优先级，非安全可调用的安全优先级次之，非安全属性的安全优先级最低。默认情况下，任何未定义的区域都是安全的

TrustZone®安全被激活之后，默认安全状态如下：

- CPU: Cortex®-M33 在复位后处于安全状态。启动地址必须在安全区域中。
- 内存映射：SAU 在复位后是完全安全的。整个内存映射是完全安全的。最多有 8 个 SAU 可配置区域用于安全属性。
- Flash 存储器：
  - Flash 安全区域是由安全水印用户选项定义的。所有闪存都是完全安全的。
  - Flash 块的功能在复位后是非安全的。即使所有的闪存通过 IDAU/SAU 和闪存安全水印选项字节配置都是非安全的，也可以使用基于闪存块的特性配置易失性安全区域：使用 Flash 块的配置寄存器，任何页面都可动态编程为安全模式。
  - SRAM：所有 SRAM 在复位后是安全的。基于内存保护块的控制器（MPCBB）是安全的。
- 非安全内存视图与其他 Cortex®-M 内核相同。
- 安全内存空间分为两种内存类型：
  - 安全：包含安全程序代码和数据，如栈区和堆区。
  - 非安全可调用（NSC）：包含入口函数（例如 API 的入口点），这是为了防止非安全应用程序分为无效入口点。

5 调试模式

5.1 侵入式调试

侵入式调试定义为一种调试过程 - 用户控制并观察处理器活动。大多数调试功能被认为是侵入式调试，因为它们允许用户停止处理器并修改其状态。

DBGEN 和 SPIDEN 控件都具有侵入式调试权限。

5.2 非侵入式调试

非侵入式调试定义为一种调试过程 - 用户观察处理器但不进行控制。嵌入式跟踪宏单元™（ETM）接口和性能监视器寄存器是非侵入式调试的特性。

NIDEN 和 SPNIDEN 控件都具有非侵入式调试权限。允许侵入式调试的时候总是允许非侵入式调试。

6 调试访问

6.1 安全调试访问

安全调试访问在所有内存区域和器件外设范围内提供了对所有指令执行的完全可见性。它允许跟踪和调试运行在目标上的安全和非安全软件。

仅在此模式下才能调试安全固件。

在安全状态下运行的代码可以访问安全和非安全信息。

6.2 非安全调试访问

非安全调试视图保护安全内存和外设。在非安全模式下，这些对于调试器是不可见的。调试和跟踪功能仅限于非安全系统资源。

7 Flash 存储器保护

7.1 TrustZone®被禁用后的读出保护级别

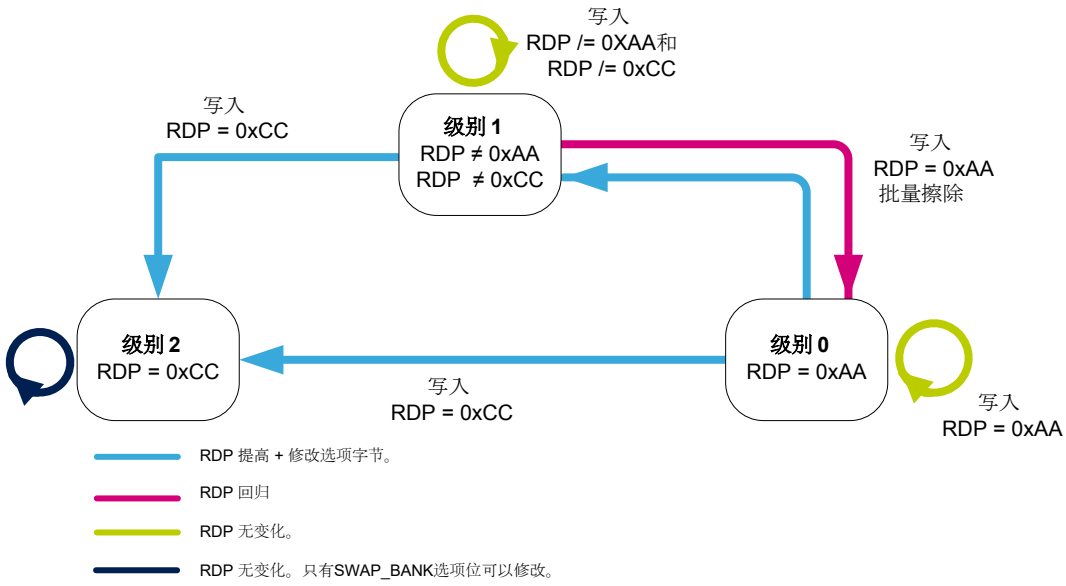
有三个读出保护级别，如下所示：

- 级别 0：所有对用户闪存的读取/编程/擦除操作被允许。
- 级别 1：针对调试器和存储在 RAM 中的潜在恶意代码，对闪存内容进行保护，无法读出。
- 级别 2：所有的调试功能被禁用，不能再从 SRAM 和系统内存启动。

7.2 TrustZone®被禁用后的 RDP 级别转换流程

TZEN 被清除后的 RDP 级别转换流程在图 2 中加以说明。

图 2. TrustZone®被禁用（TZEN = 0）后的 RDP 级别转换流程



7.3 TrustZone®启用后的读出保护级别

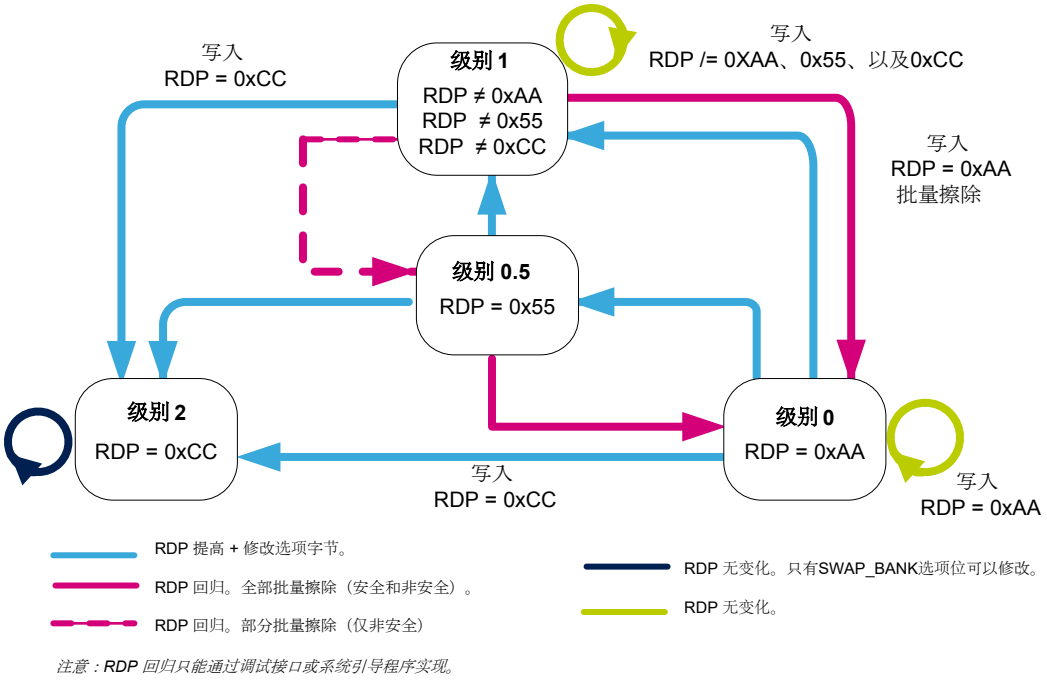
除了前面提到的 RDP 级别设置，有一个新的 RDP 级别名为 0.5，它允许以下功能：

- 所有对非安全闪存的读写操作都是可行的。禁止对安全区域的调试访问。仍然可以调试访问非安全区域。
- 非安全调试模式：当 CPU 处于非安全状态时，可以进行非安全调试。

7.4 当 TrustZone®启用后，RDP 级别转换流程

设置了 TZEN 后的 RDP 级别转换流程在图 3 中加以说明。

图 3. TrustZone®被禁用（TZEN = 1）后的 RDP 级别转换流程



注意：RDP 回归只能通过调试接口或系统引导程序实现。

8 从安全/非安全项目开始

EWARM 和 MDK-ARM 提供相似的方法来支持 STM32L5 系列微控制器。这是通过两个独立的项目（安全和非安全）完成的。

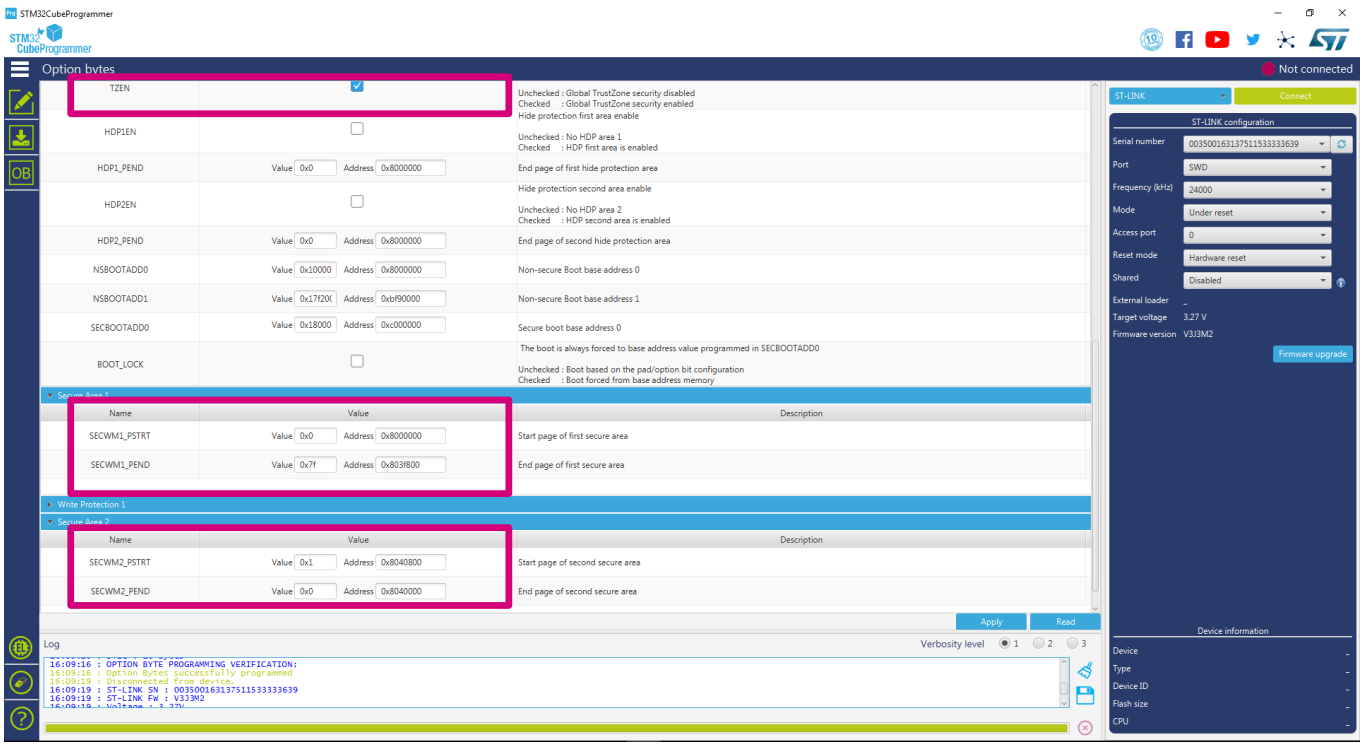
- 第 9 节 提供 MDK-ARM 项目说明。
- 第 10 节 提供 EWARM 的说明。
- 第 11 节 提供 CubeIDE 的说明。

每个部分都提供逐步说明，以使用 STM32L5 系列微控制器讲解安全和非安全部分的项目设置。

首先，使用 STM32CubeL5 包（STM32Cube_FW_L5）中的模板，它由两个子项目组成：一个用于安全应用程序部分，另一个用于非安全应用程序部分。

在开始之前，必须使用 STM32CubeProgrammer 设置选项字节，详情请参见项目的 readme.txt。该工具可从 *www.st.com* 下载，并在图 4 中加以说明。

图 4. 使用 STM32CubeProgrammer 配置选项字节



9 将 MDK-ARM 用于带 Trust Zone 的 Cortex®-M33

最新版本的 MDK-ARM（Keil®）可从官方 Arm® Keil®网站下载。MDK-ARM (Keil®)默认安装在 PC 本地硬盘上的 C:\Keil 目录下；安装程序将在开始菜单中创建 µVision® 5 快捷方式。

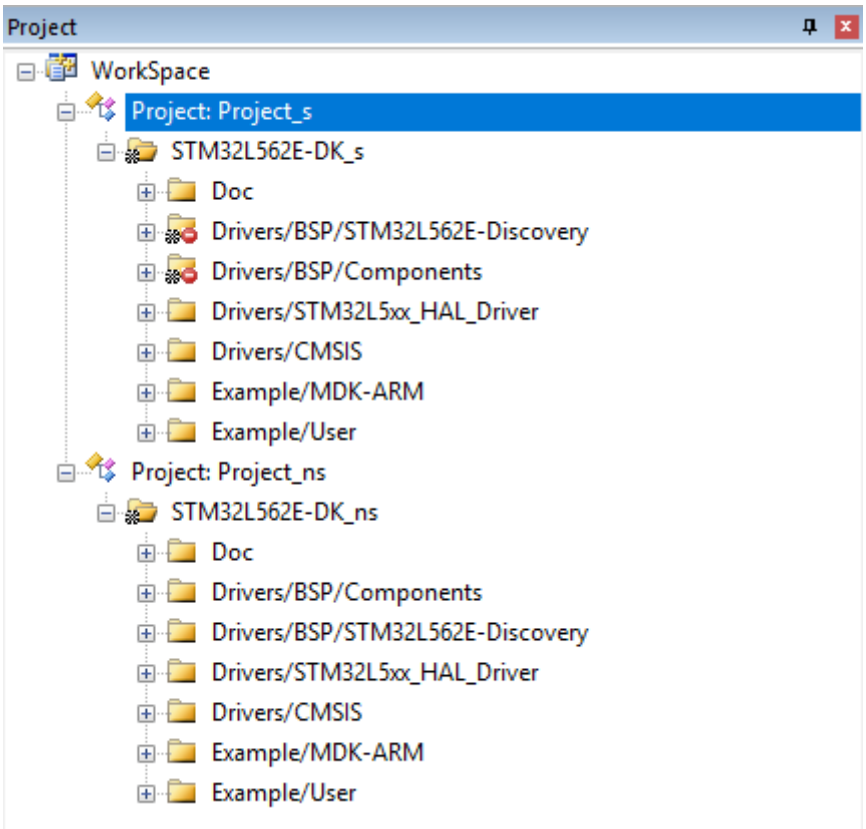
本节使用 MDK-ARM v5.27.0.0 和 STM32L562-DK 探索板。

9.1 安全项目设置

本节概述安全项目设置。

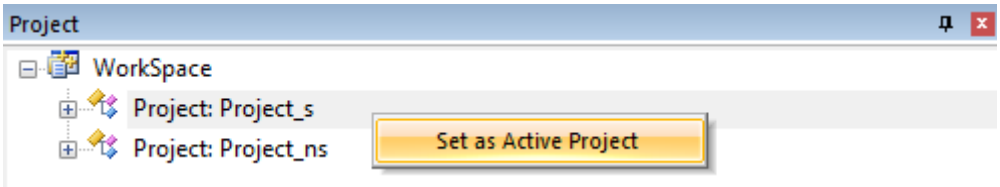
1. 打开多项目工作区文件“Project.uvmpw”，该文件允许用户同时处理两个项目。打开的项目显示在项目浏览器中，如图 5 中所示。

图 5. **MDK-ARM 项目结构**



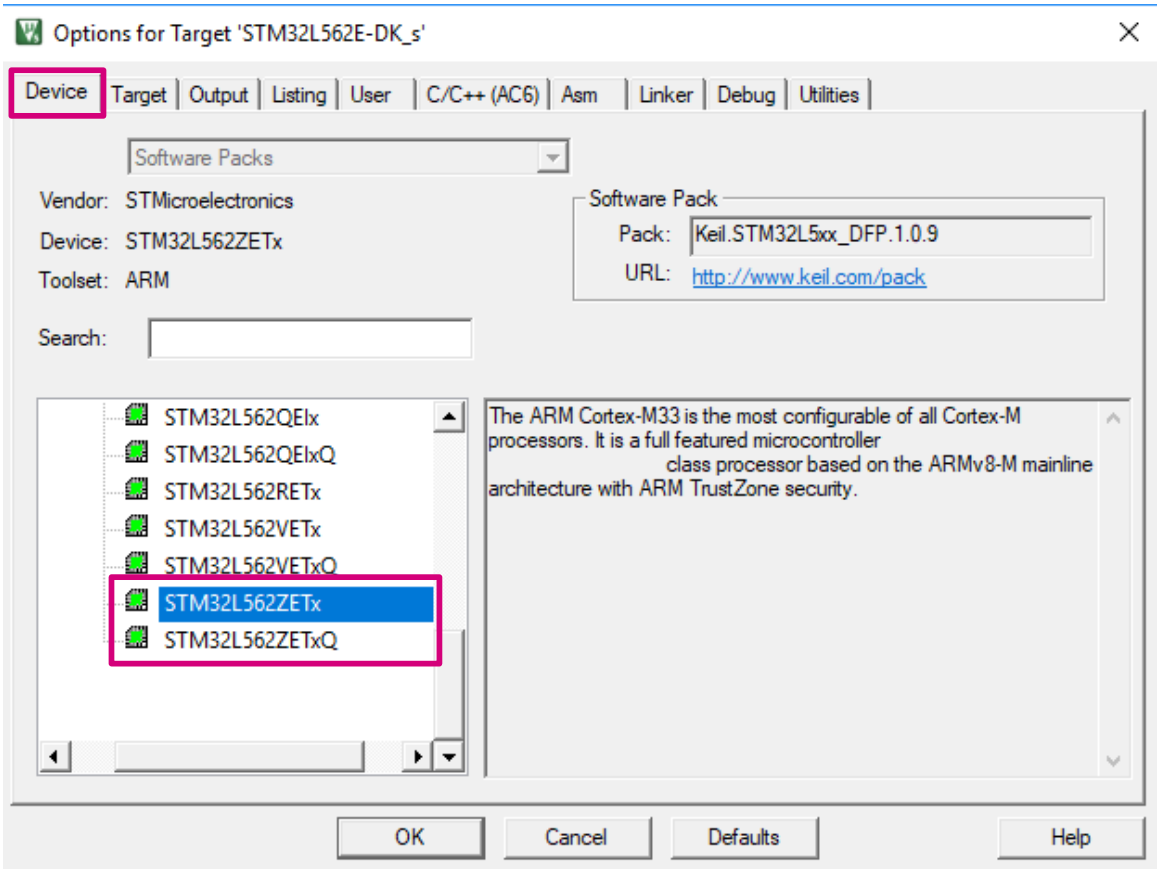
2. 将 project_s 设为活动项目，参见图 6。

图 6. **选择安全项目**



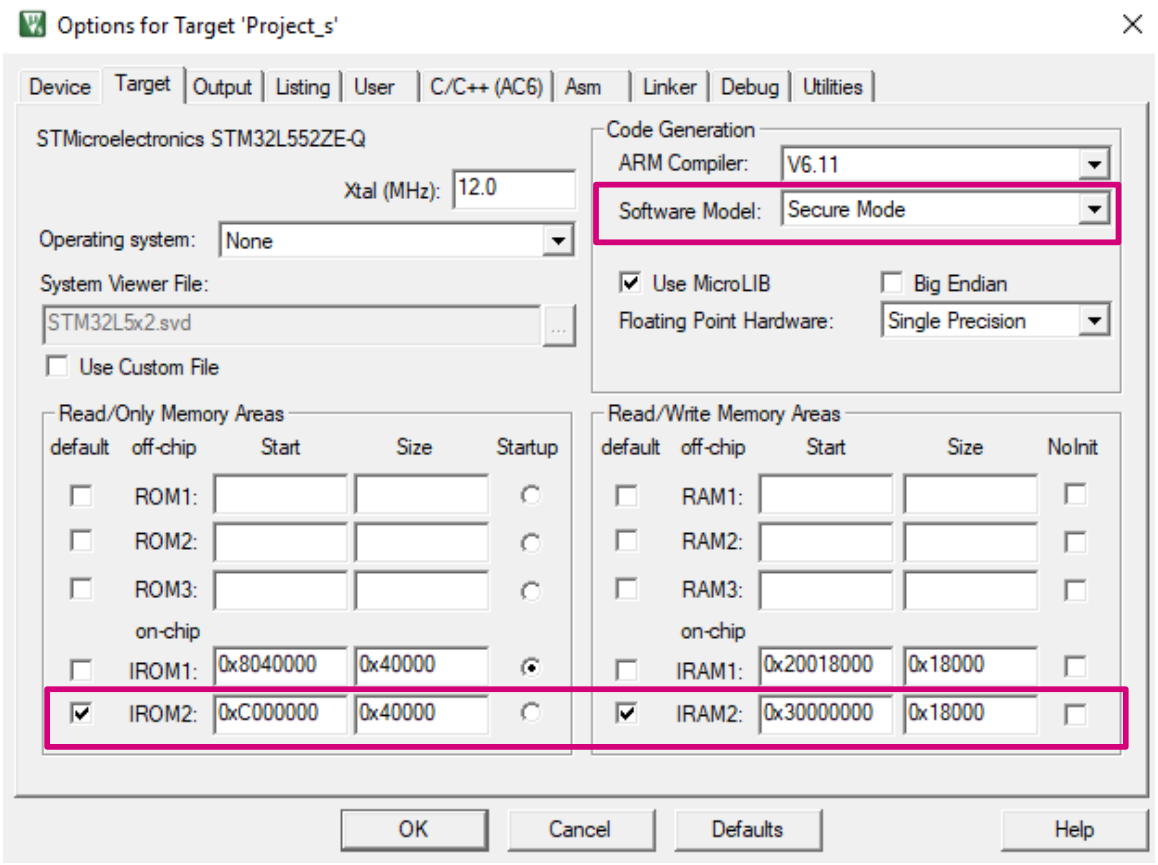
3. 选择正确的器件，方法是：打开配置窗口，然后选择： Project / Options for Target / Device，接下来从列表中选择器件（参见图 7）。

图 7. 设备选择



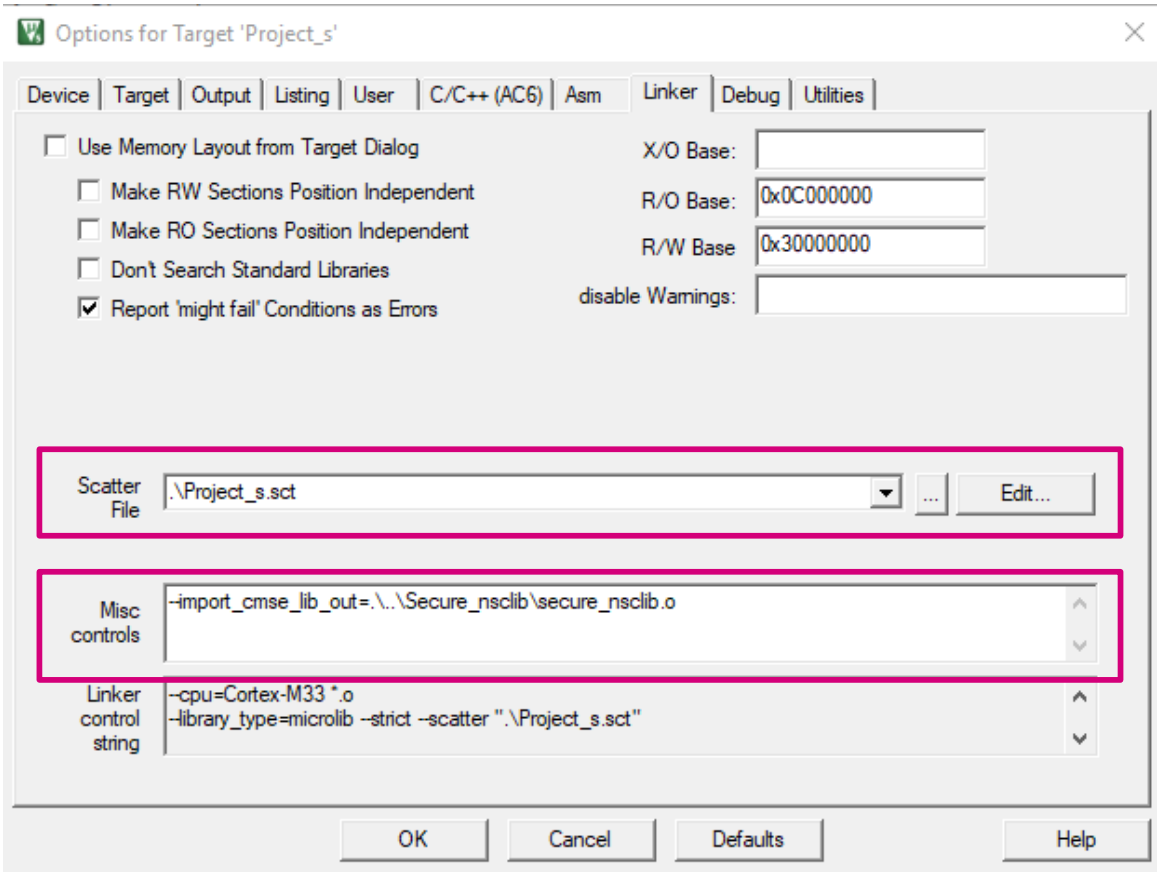
4. 从 Project / Options for Target / Target / Code Generation 部分选择“Software Model（软件模型）”为“Secure（安全）”。确保选择正确的内存区域。请参见图 8：
   – 安全启动地址：位于 0x0C000000 的闪存：安全闪存
   – 安全启动地址：位于 0x30000000 的 SRAM1：安全 SRAM

图 8. **Project_s 目标选项**



5. 确保安全的非安全可调用函数（NSC）对象文件“secure_nsclib.o”在 Misc Controls 部分下的 Project / Options for Target / Linker 中定义。
使用[--import_cmse_lib_out ..\lib\nsclib_Secure.o]命令创建输出库：nsclib_Secure.o。
该文件是在编译安全项目期间自动生成的，它包含所有使用前缀
__attribute__((cmse_nonsecure_entry))声明的非安全可调用函数。
请参见图 9。

图 9. Project_s 链接器配置



在分散加载文件部分下，检查该文件是否包含正确的地址，如图 9 中所示。
该文件由链接器使用，并决定内存布局的组织方式。分散加载文件示例在图 10 中给出。

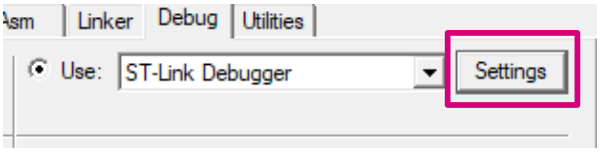
图 10. 分散加载文件示例

Project_s.sct

```
; *************************************************************
; *** Scatter-Loading Description File generated by uVision ***
; *************************************************************

LR_IROM2 0x0C000000 0x00040000  {    ; load region size_region
  ER_IROM2 0x0C000000 0x0003E000  {  ; load address = execution address
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
   .ANY (+XO)
  }
  RW_IRAM2 0x30000000 0x00018000  {  ; RW data
   .ANY (+RW +ZI)
  }
}

LR_IROM3 0x0C03E000 0x00002000  {    ; load region size_region
  ER_IROM3 0x0C03E000 0x00002000  {  ; load address = execution address
   *(Veneer$$CMSE)                   ; check with partition.h
  }
}

```

6. 从 Project / Options for Target / Debug 选择“ST-LINK Deb ugger”为调试器。请参见图 11。

图 11. 目标选项调试



如果“ST-LINK Debugger”没有显示在列表中：

a. 前往 C:\Keil install directory

b. 打开 TOOLS.INI 文件并进行以下更改：

   i. 查找[ARMADS]：

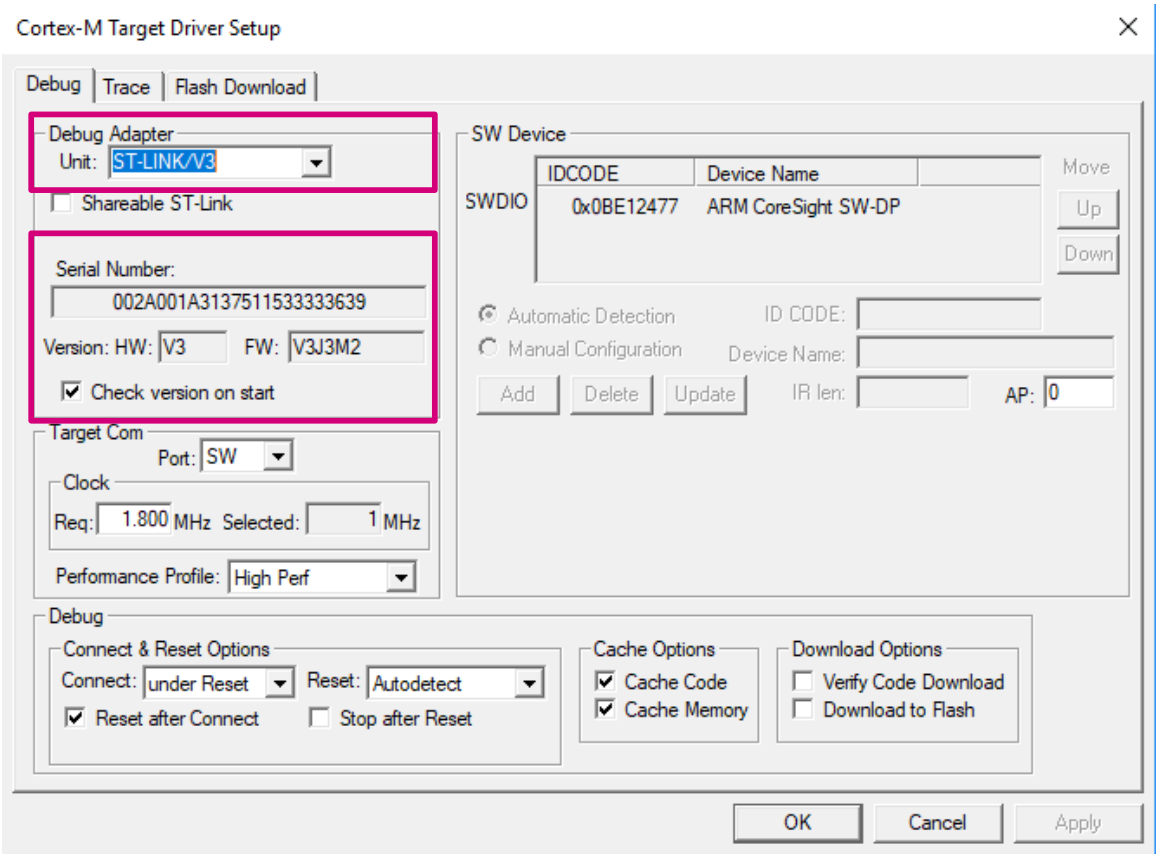
      所有基于 Armv8M 的器件都需要处理器 SARMV8M.DLL。TOOLS.INI 文件包含 CPUDLL3 = SARMV8M.DLL（TDRV2、TDRV13、TDRV14、TDRV15、TDRV16）。

      在本例中，ST-Link 驱动程序注册为 TDRV6，且可能根据项目 TDRV6=STLink\ST-LINKIII-KEIL_SWO.dll （“ST-Link Debugger”）的不同而变化。

   ii. 添加 TDRV6 到 CPUDLL3= SARMV8M.DLL:CPUDLL3 = SARMV8M.DLL（TDRV2、TDRV6、TDRV13、TDRV14、TDRV15、TDRV16）中的列表。

7. 从“Debug”设置选项卡确保调试器已连接，如图 12 中所示。

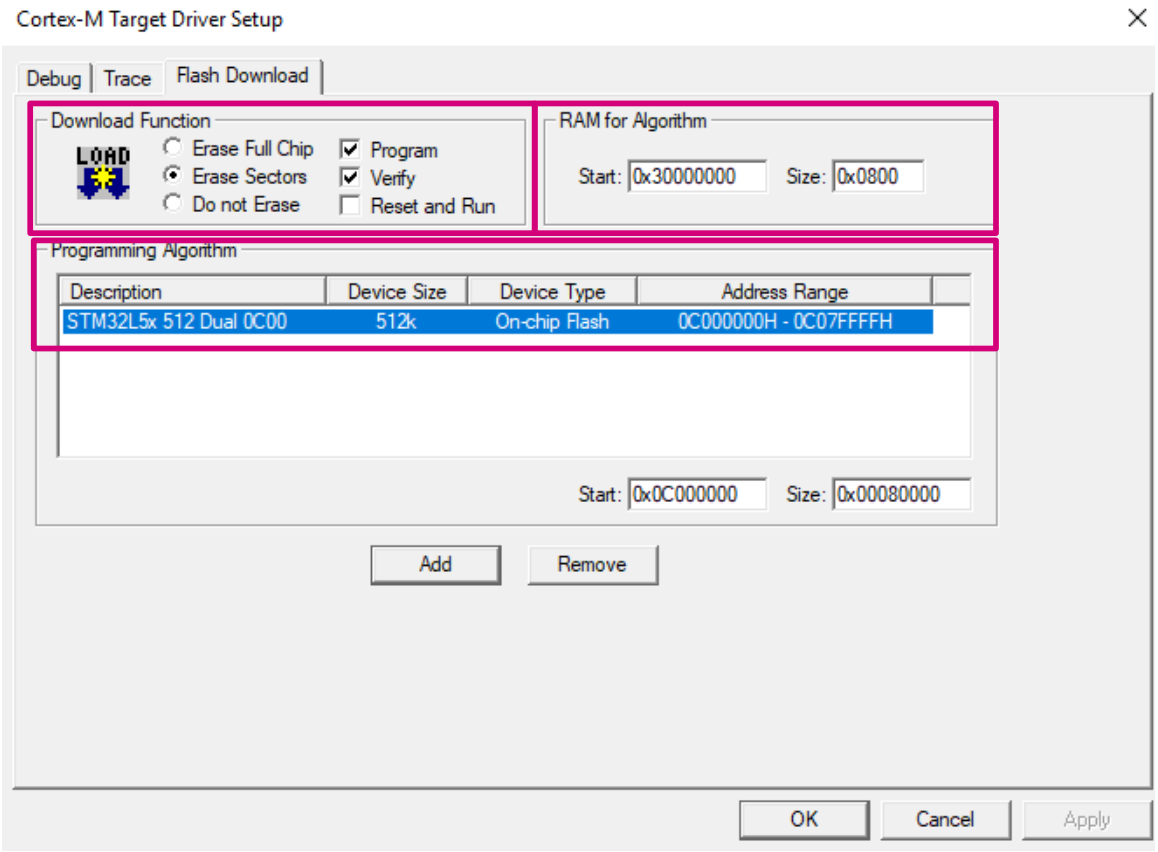
图 12. 调试配置



从“Flash Download”选项卡选择正确的闪存加载程序（参见图 13）：

– “Download Function（下载功能）”：设置 Flash 操作。
– 用于算法的 RAM：定义用于载入和执行编程算法的地址空间。通常，地址空间位于片上 RAM 中。
– “Program Algorithm（编程算法）”：包含编程 Flash 的定义。

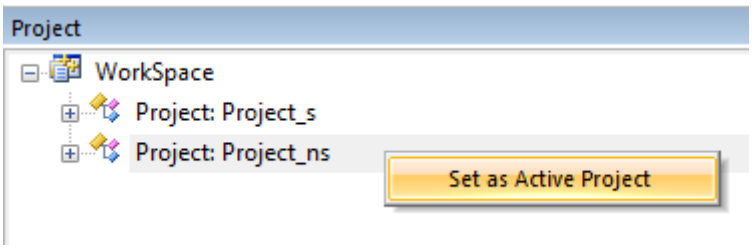
图 13. 闪存加载程序设置



9.2 非安全项目设置

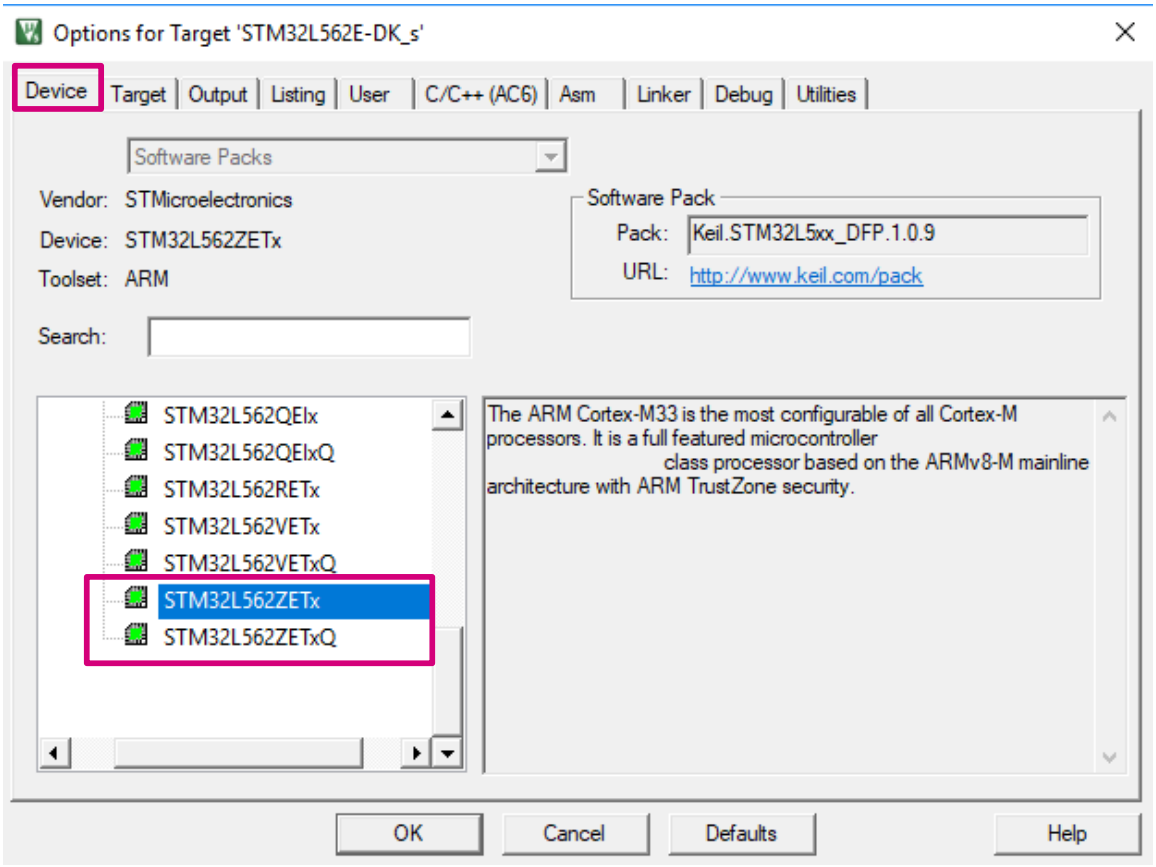
1. 设置 project_ns 活动项目（参见图 14）。

图 14. 选择 Project_ns 非安全项目



2. 打开配置窗口以选择正确的器件：选择 Project / Options for Target（参见图 15）。

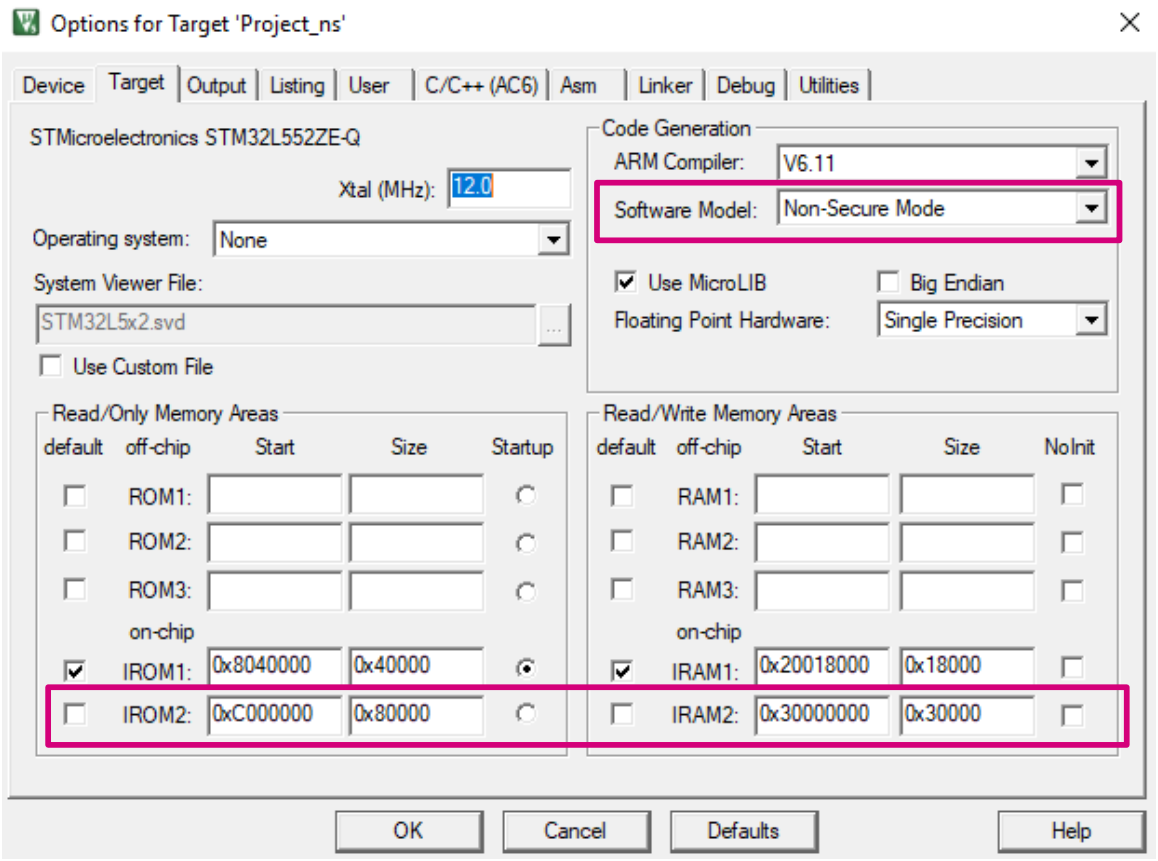
图 15. 设备选择



3. 确保从 Project / Options for Target / Target 选择正确的内存区域：
   – 启动地址 0：位于 0x08040000 的闪存：非安全闪存
   – 启动地址 1：位于 0x20018000 的 SRAM：非安全 SRAM

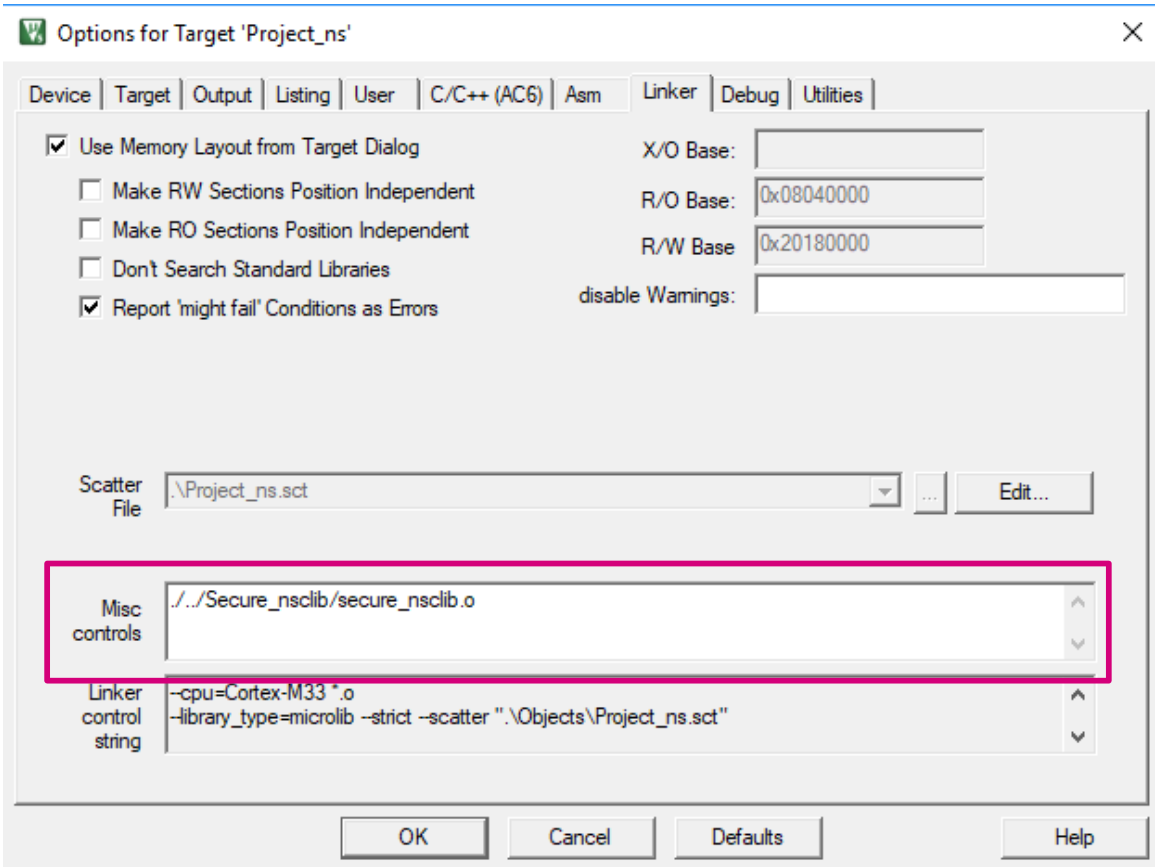
   软件模型必须在非安全模式下设置（参见图 16）。

图 16. 内存配置



4. 添加来自安全项目的导入库：此文件在链接时自动包含在非安全项目中。它允许非安全部分调用安全部分的函数（参见图 17）。

图 17. 链接器选项



在分散加载文件部分下，检查该文件是否包含正确的地址，如图 17 中所示。
该文件由链接器使用，并决定内存布局的组织方式。分散加载文件示例在图 18 中给出。

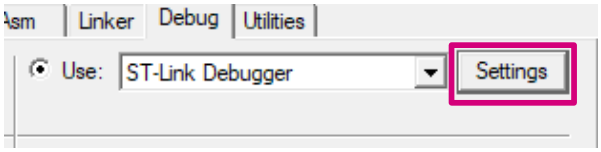
图 18. 分散加载文件示例

Project_ns.sct

```
1  ; *************************************************************
2  ; *** Scatter-Loading Description File generated by uVision ***
3  ; *************************************************************
4
5  LR_IROM1 0x08040000 0x00040000  {    ; load region size_region
6    ER_IROM1 0x08040000 0x00040000  {  ; load address = execution address
7     *.o (RESET, +First)
8     *(InRoot$$Sections)
9     .ANY (+RO)
10    .ANY (+XO)
11   }
12   RW_IRAM1 0x20018000 0x00018000  {  ; RW data
13    .ANY (+RW +ZI)
14   }
15 }
```

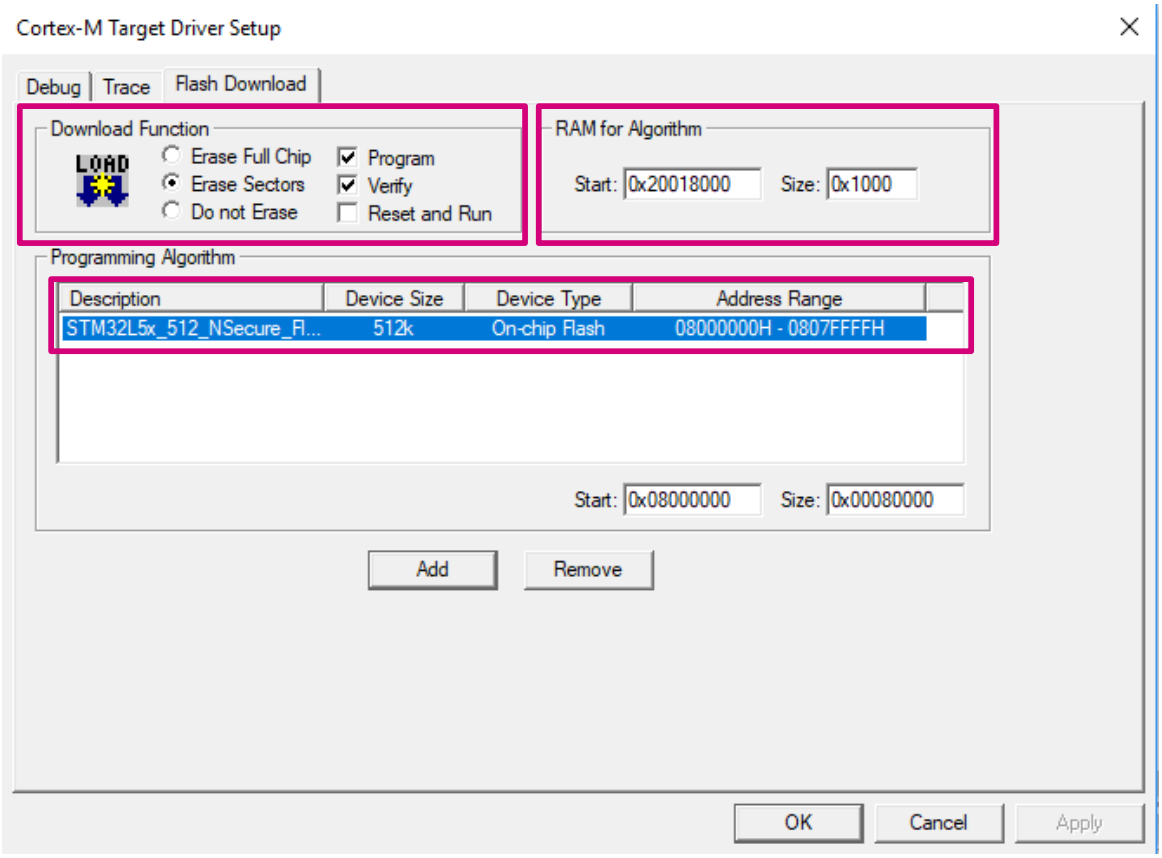
5. 从 Project / Options for Target / Debug 选择“ST-LINK debugger”（参见图 19）。

图 19. 调试设置



6. 从 Debug settings / Flash Download 窗口（参见图 20）选择：
   – 下载功能：设置 Flash 操作
   – 用于算法的 RAM：定义用于载入和执行编程算法的地址空间。通常，地址空间位于嵌入式 RAM 中。
   – 编程算法：包含编程 Flash 的定义。

图 20. FlashLoader 配置



9.2.1 编译项目

现在可以同时编译两个项目。从 Project / Batch Setup（项目/批设置）（参见图 21 和图 22）或从菜单栏中可用的图标转到“批设置”菜单并选择这两个项目。

提示 *必须首先编译安全项目，以便为非安全项目创建导入库。为了在编译非安全项目之前编译安全项目，它必须在编译顺序中处于首位。*

图 21. 项目批设置

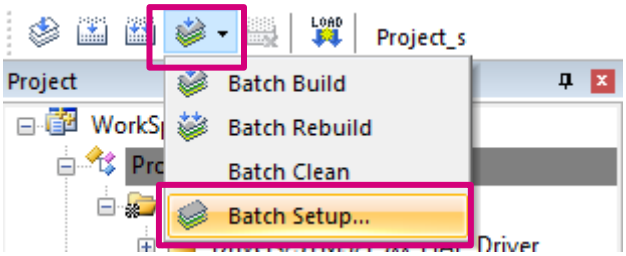
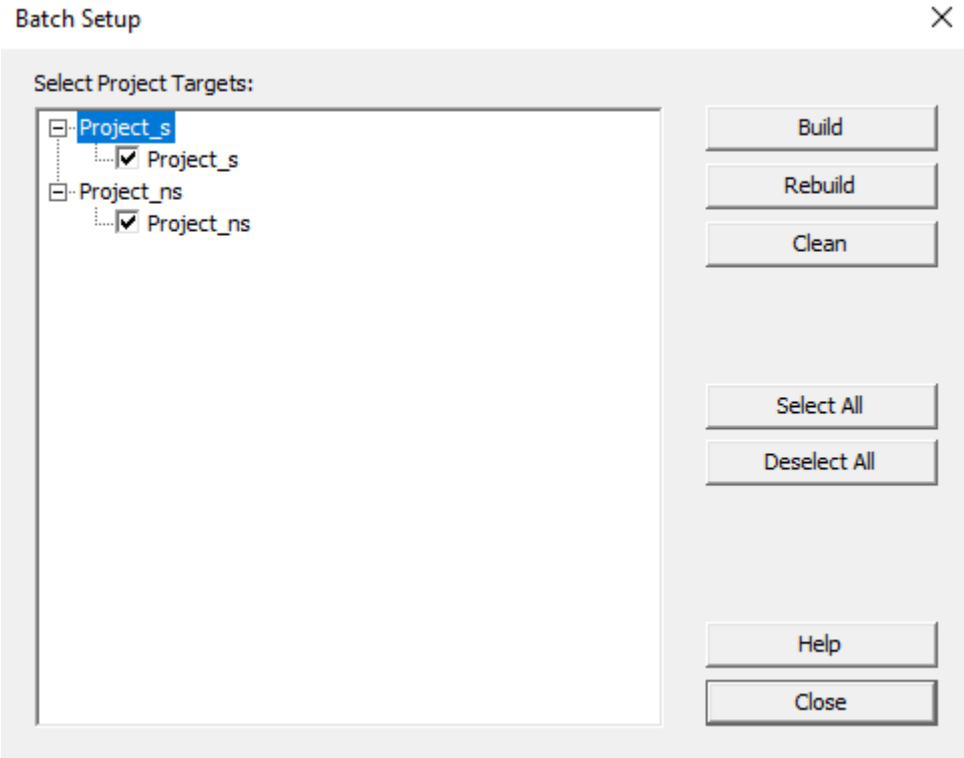
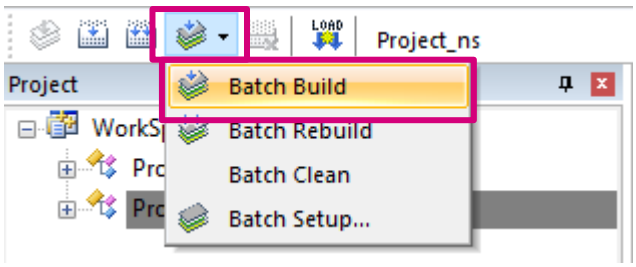


图 22. 项目编译顺序



然后，从同一个菜单中单击“Batch Build（批编译）”来编译这两个项目（参见图 23）。

图 23. 在一个步骤中编译两个项目

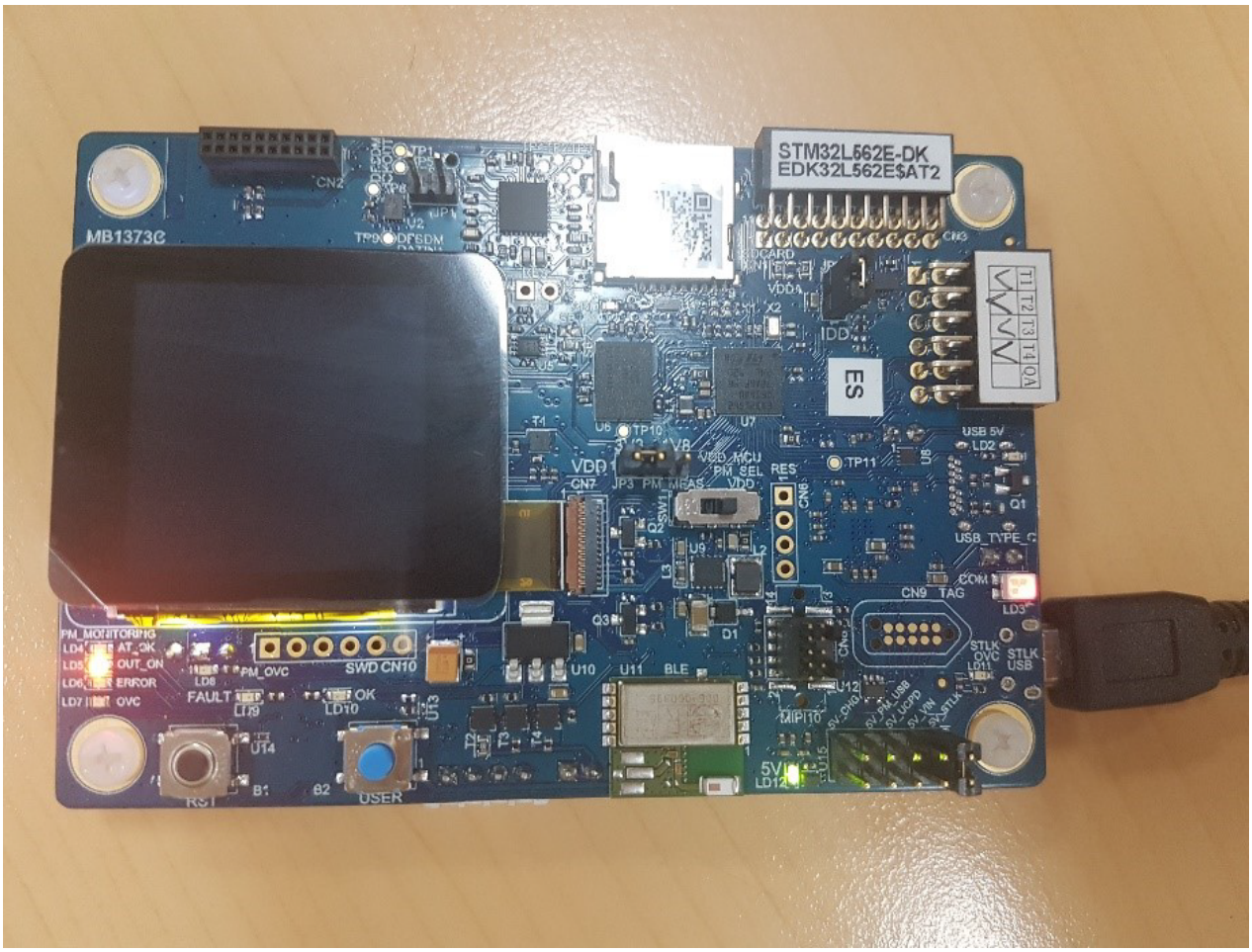


9.3 从安全代码执行到非安全代码

下载项目前，必须连接 STM32L562E-DK 探索板，连接方式如下：

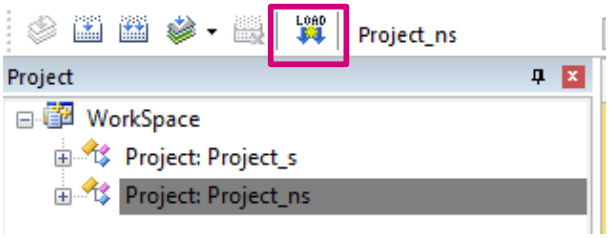
1. 连接探索板上的 ST-LINKV3 编程和调试工具，方法是将 USB 电缆插入板件 CN17（ST-LINK USB 链接器）。ST-LINKV3 连接好之后，LD3 亮起为红色，如图 24 中所示。

图 24. STM32L562E-DK 探索板处于连接状态



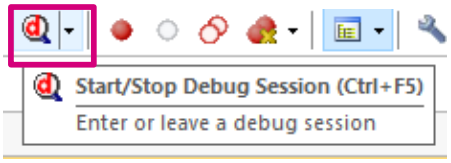
2. 选择 Project_ns 项目作为活动项目，然后加载非安全二进制代码。选择 Project_s 项目作为活动项目，然后加载安全二进制代码。图 25 对此进行了说明。

图 25. 加载非安全二进制代码



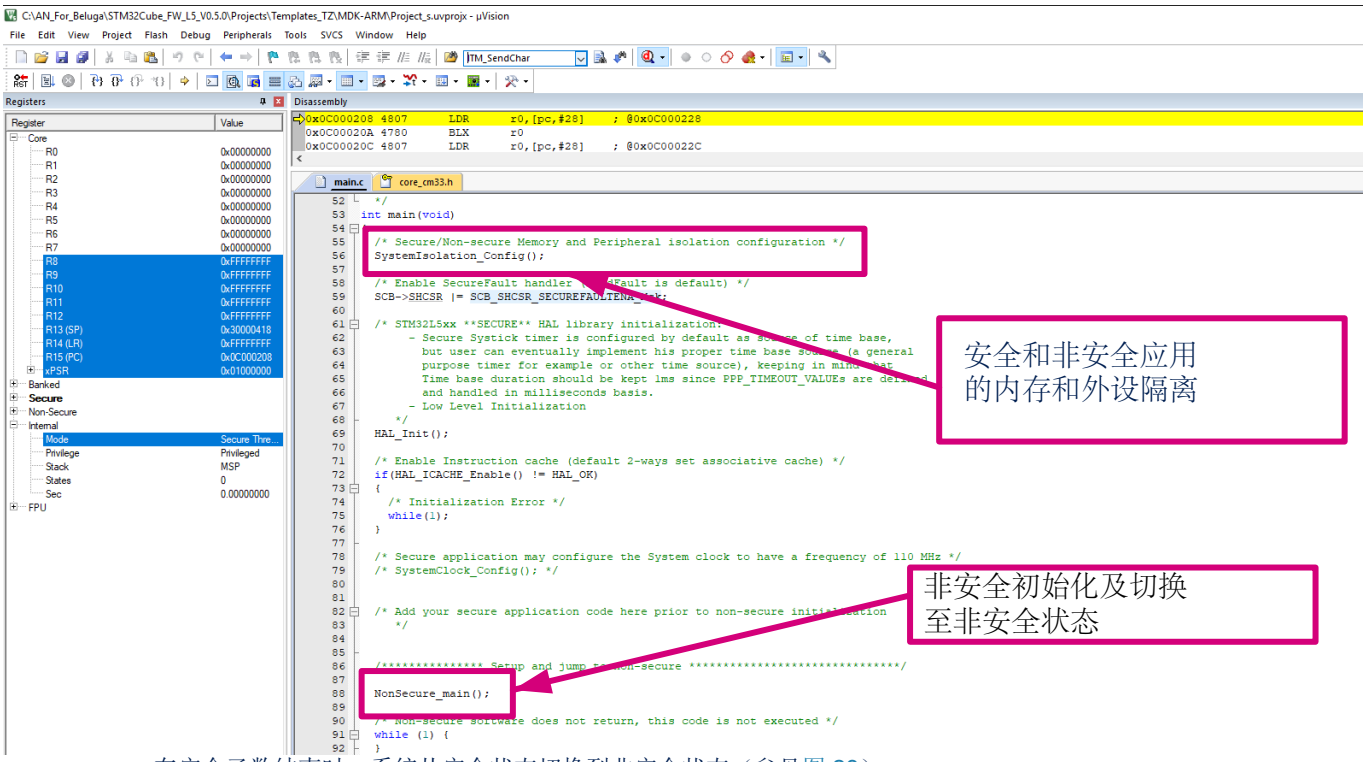
3. 单击工具栏中的“下载和调试”按钮来启动调试会话，如图 26 中所示。

图 26. 下载和调试按钮



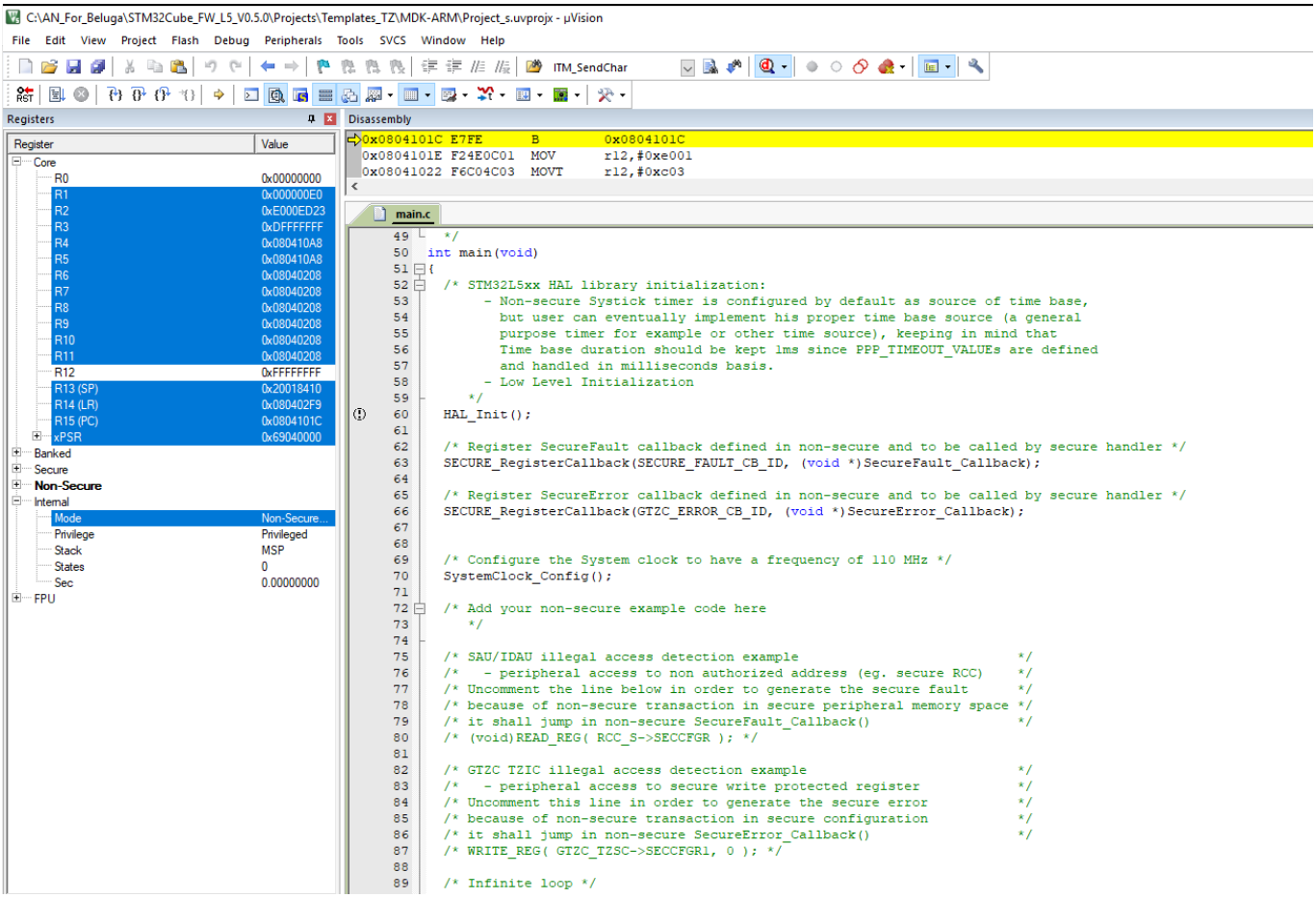
提示　　如下所示，系统总是首先在安全代码（main.c）中启动，然后安全应用程序启动非安全应用程序。

图 27. Main.c 示例代码



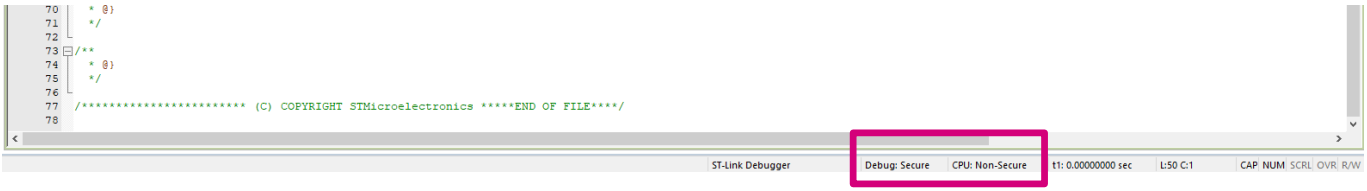
在安全函数结束时，系统从安全状态切换到非安全状态（参见图 28）。

图 28. 代码切换到非安全代码状态



安全状态由 Keil®界面底部的状态栏提供，如图 29 中所示。

图 29. CPU 状态



10 将 EWARM 用于带 Trust Zone® 的 Cortex M33

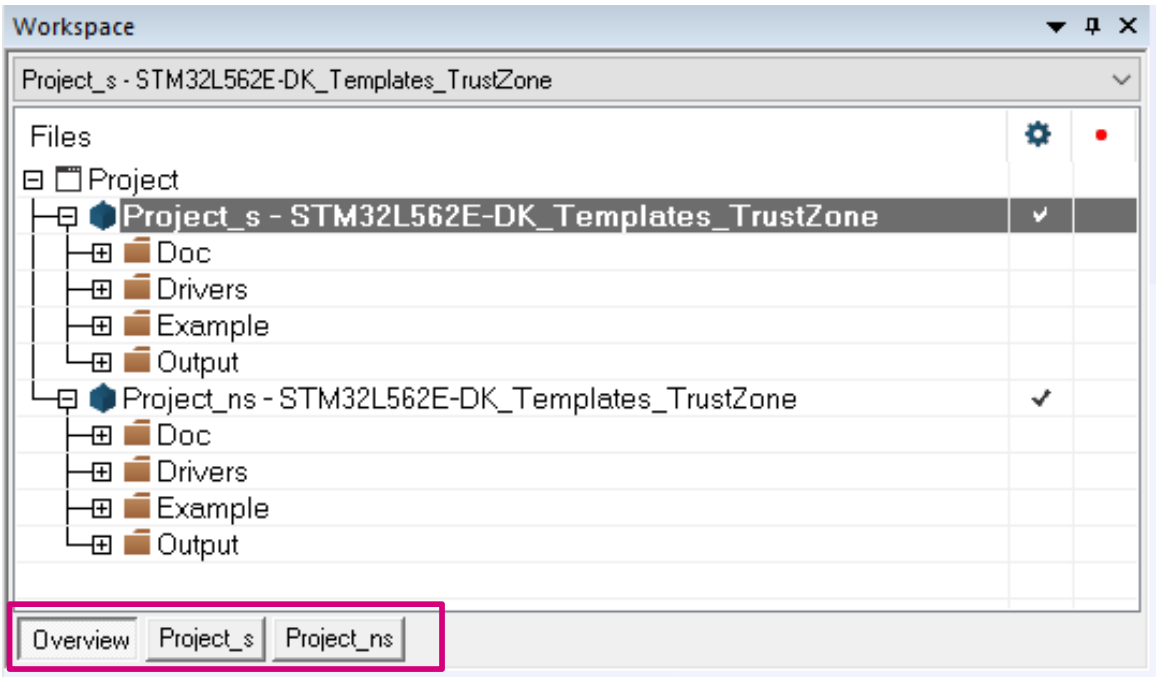
面向 Arm®（EWARM）的最新版本 IAR 嵌入式工作台可用从 IAR System 的官方网站下载。
该部分使用 EWARM v8.40.1 和 STM32L562-DK 探索板。

10.1 安全项目设置

要配置安全项目，第一步是打开“多项目”工作区文件：Project.eww，该文件允许用户同时处理两个项目。

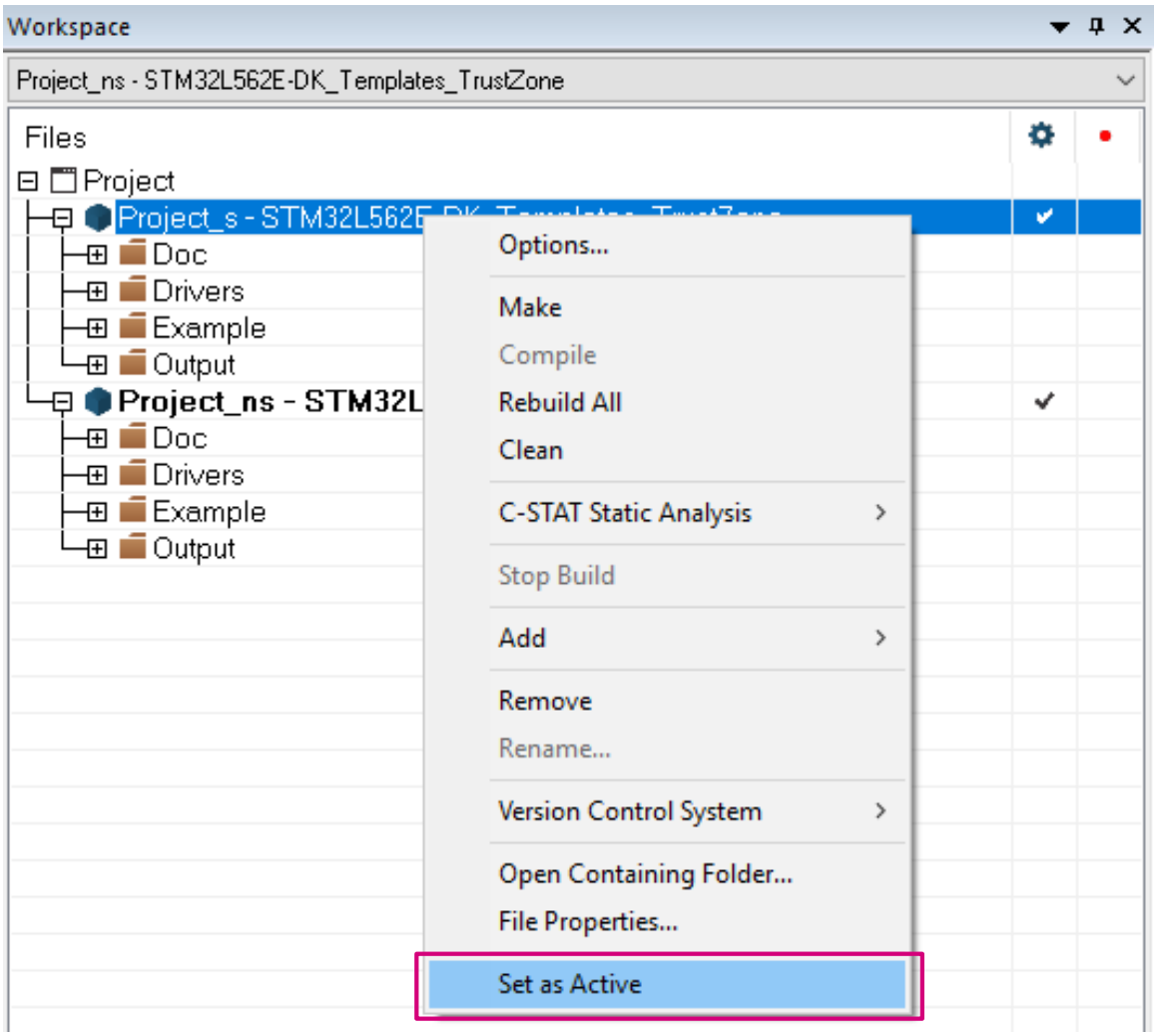
1. 打开的项目出现在“Project Explorer（项目浏览器）”视图中，如图 30 中所示。

图 30. EWARM v8.40.1 项目浏览器视图



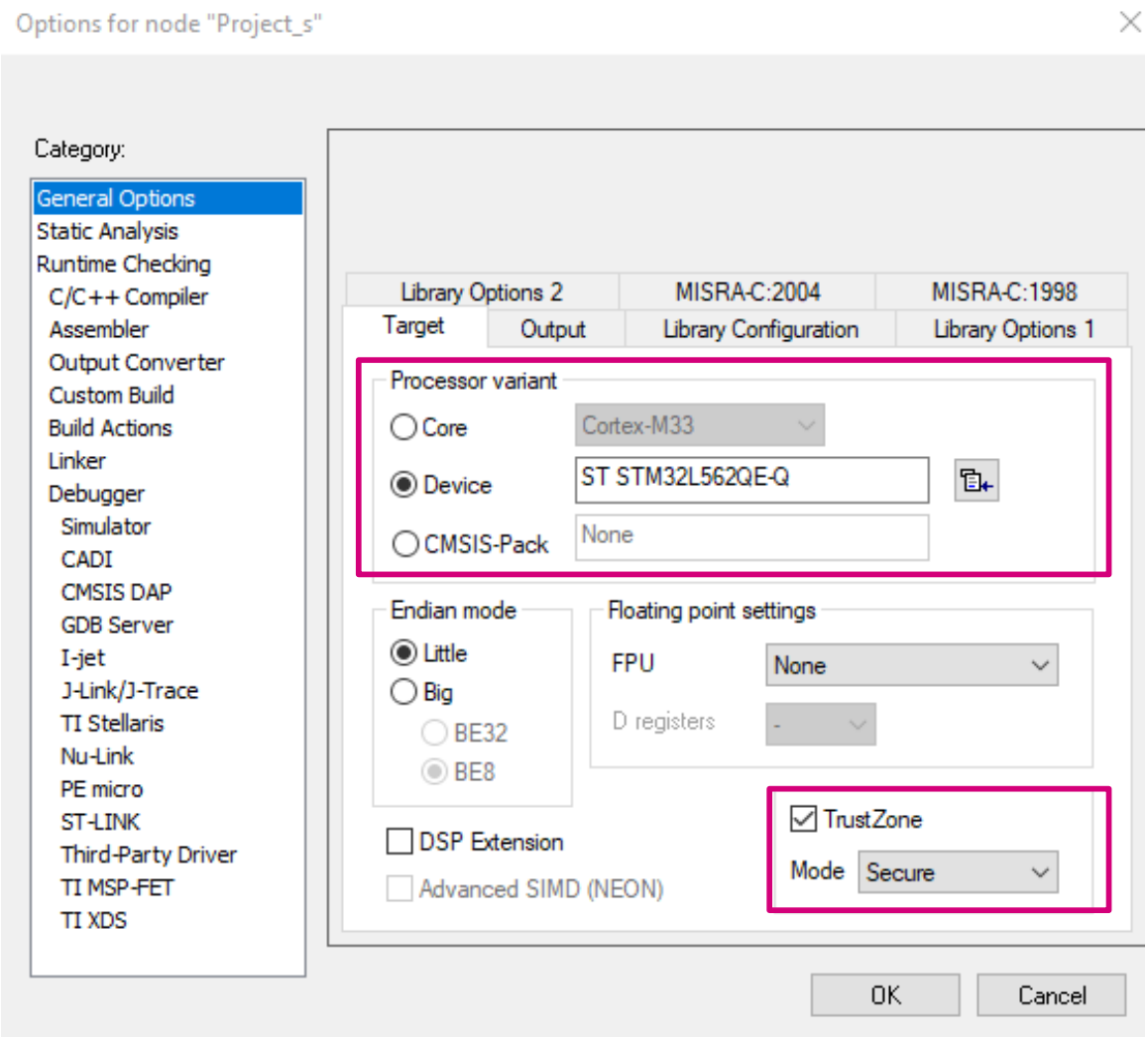
2. 将 project_s-STM32L562E-DK_Templates_TrustZone 设为活动项目，如图 31 中所示。

图 31. 将项目设为活动状态



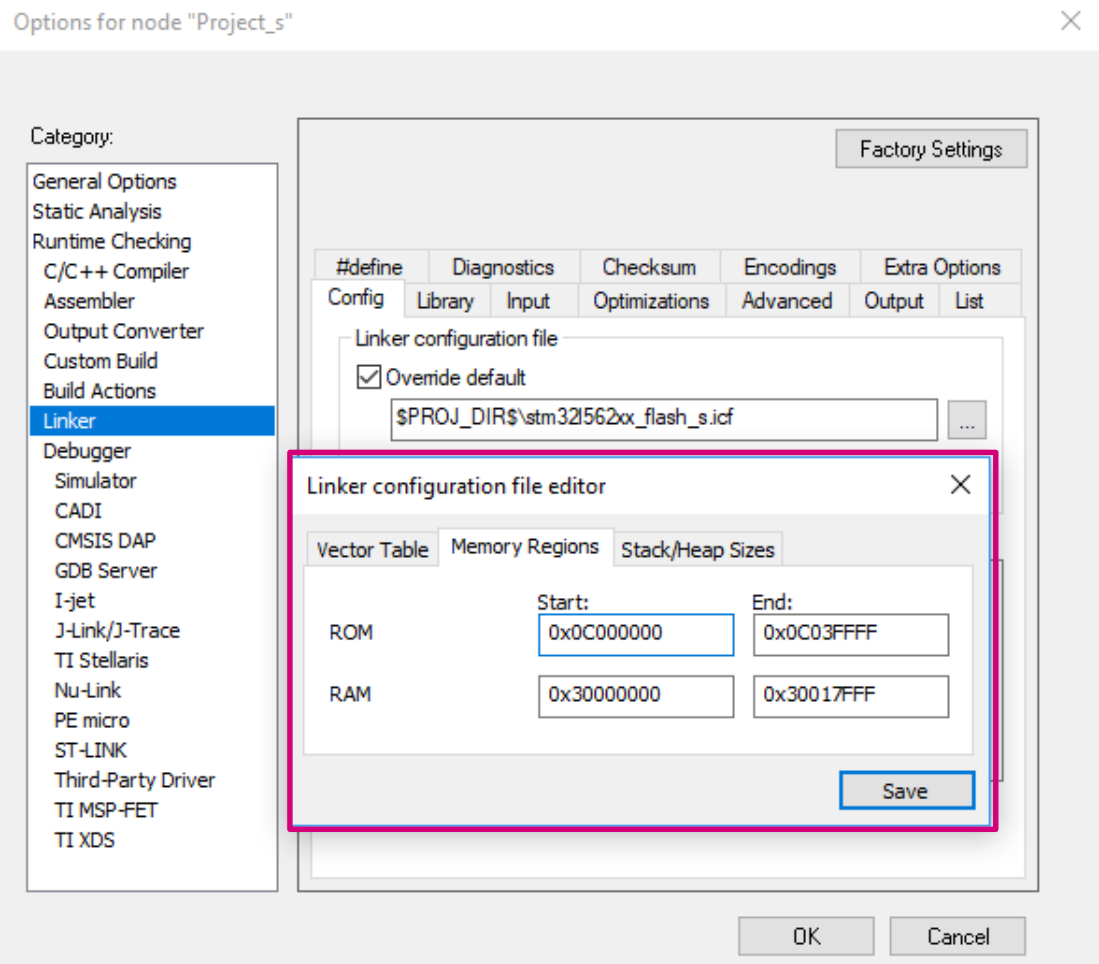
3. 选择 Project-s / Options / General Options 以打开配置窗口，然后从“Processer Variant”部分选择正确的器件。从“TrustZone”部分确保选择“安全”模式并勾选“TrustZone”复选框，如图 32 中所示。

图 32. 设备选择



4. 从 Project-s / Options / Linker / Config“链接器配置文件编辑器”部分（参见图 33）：
   a. 单击“编辑”以显示链接器配置文件编辑器。
   b. 检查链接器配置文件，以确保应用程序已链接到正确的地址：
      - 安全启动地址：Flash 位于 0x0C000000，面向安全 Flash
      - 安全启动地址：SRAM1 位于 0x30000000，面向安全 SRAM

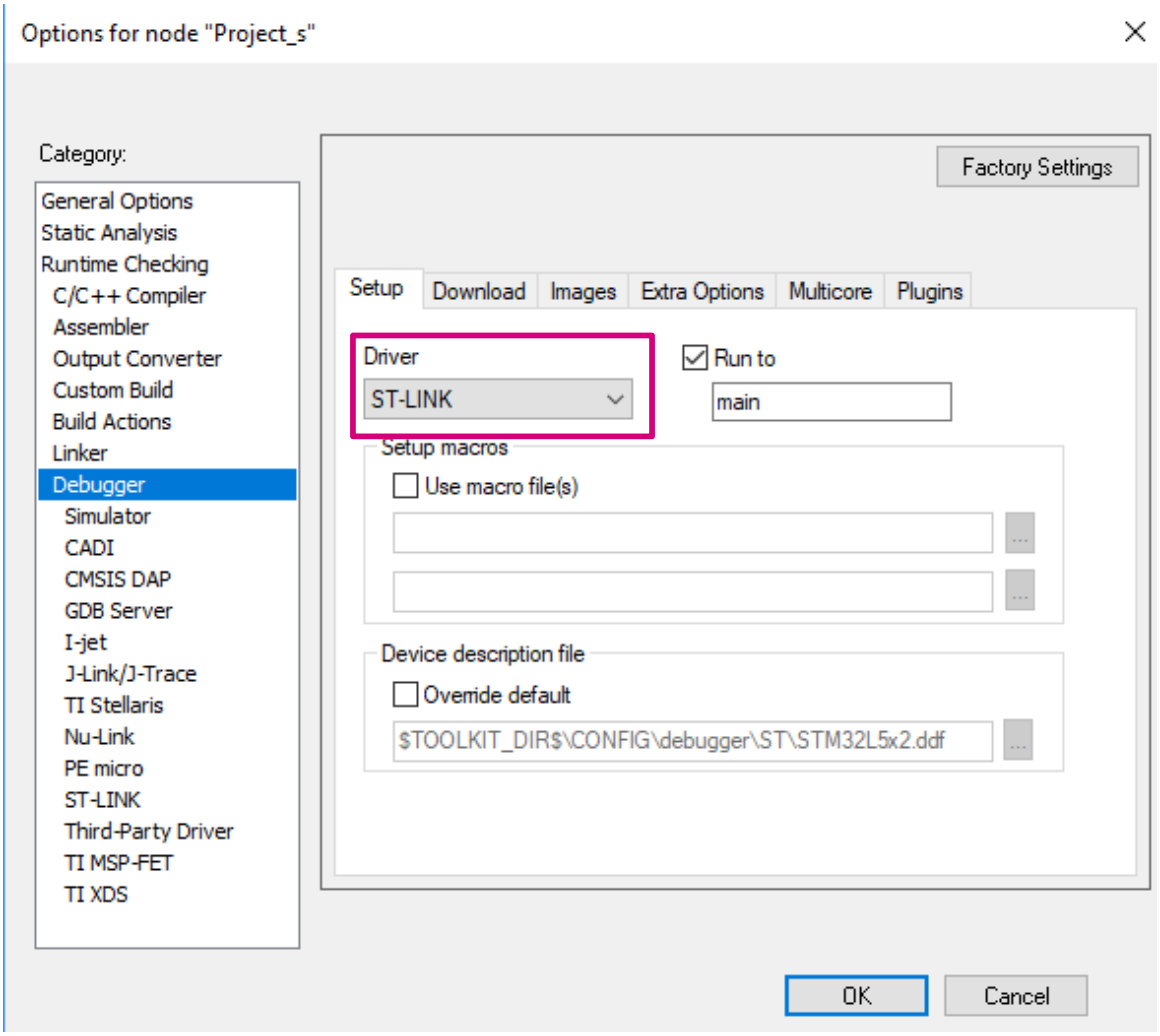
图 33. 链接器配置



该.icf 文件包含链接器需要的所有信息。

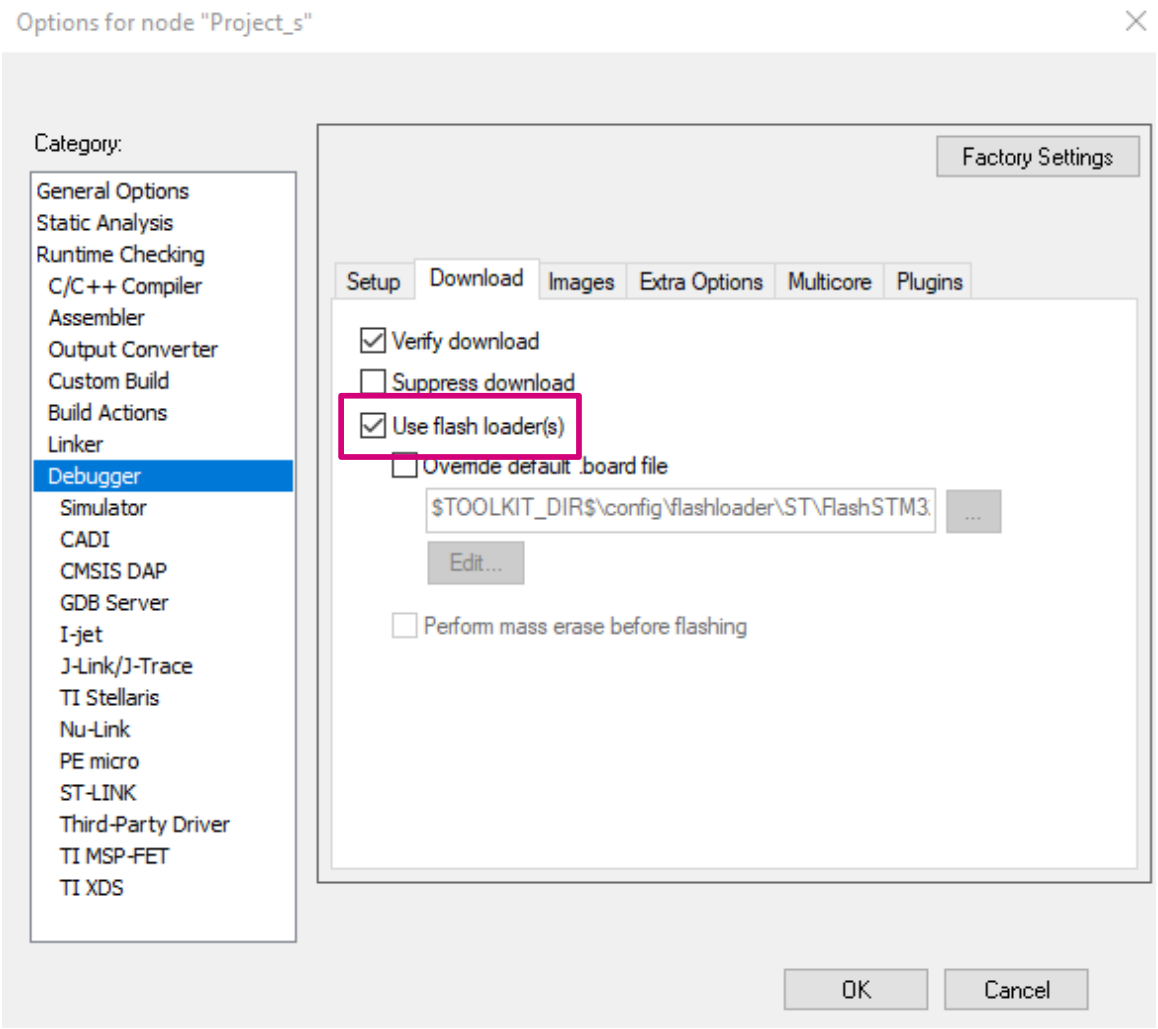
5. 从 Project / Options / Debugger（项目/选项/调试器）打开调试器选项卡。从设置部分，在驱动程序字段中选择 ST-LINK 作为调试器（参见图 34）。

图 34. 项目调试器设置



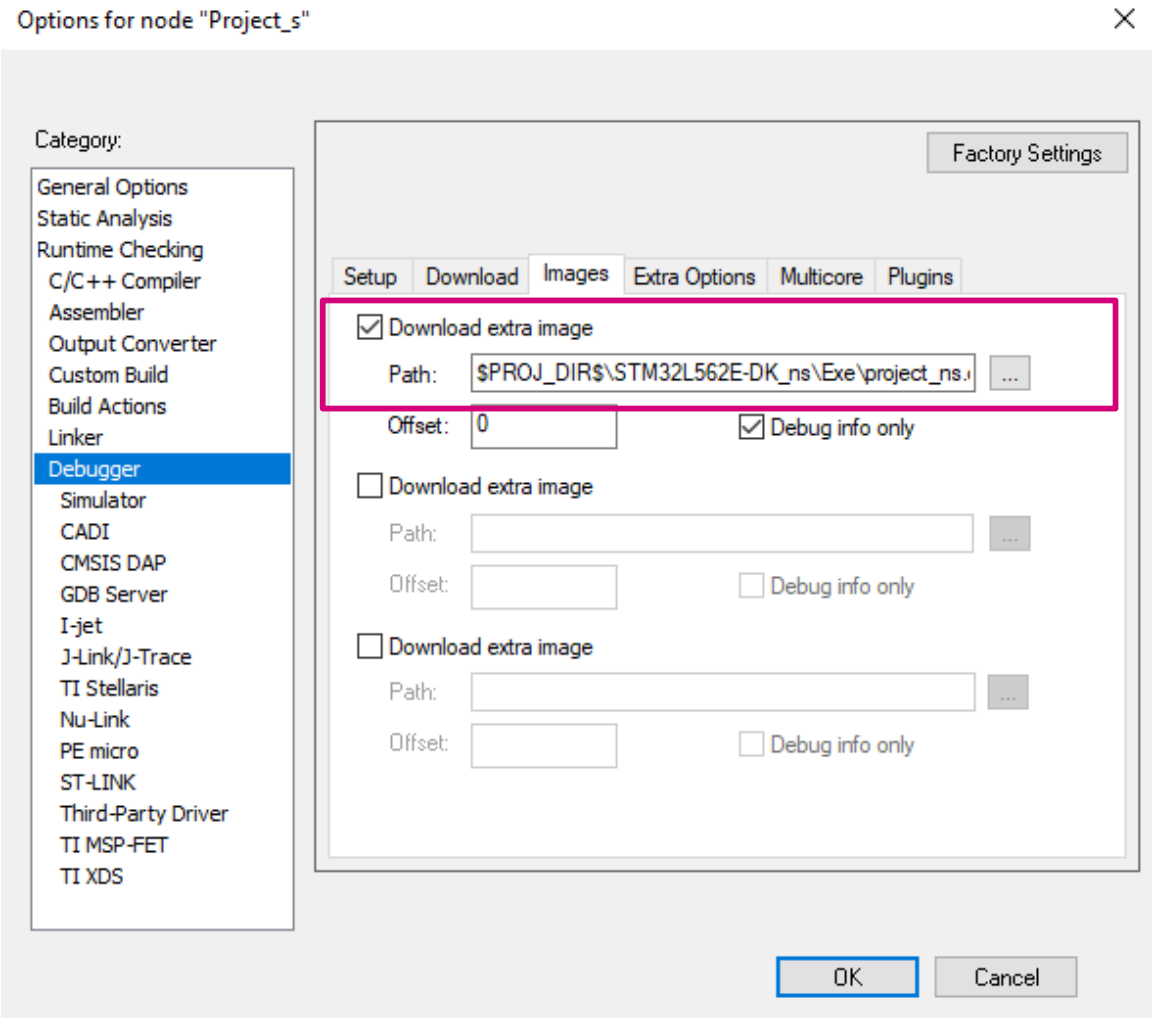
6. 从“下载”选项卡，确保“Use flash loader”已勾选（参见图 35）。

图 35. **FlashLoader 选择**



7. 安全项目必须将非安全项目输出文件指定为必须由调试器加载的额外映像。为此，请使用：Project / Options / Debugger / Images 并勾选“Download extra image”复选框（参见图 36）。

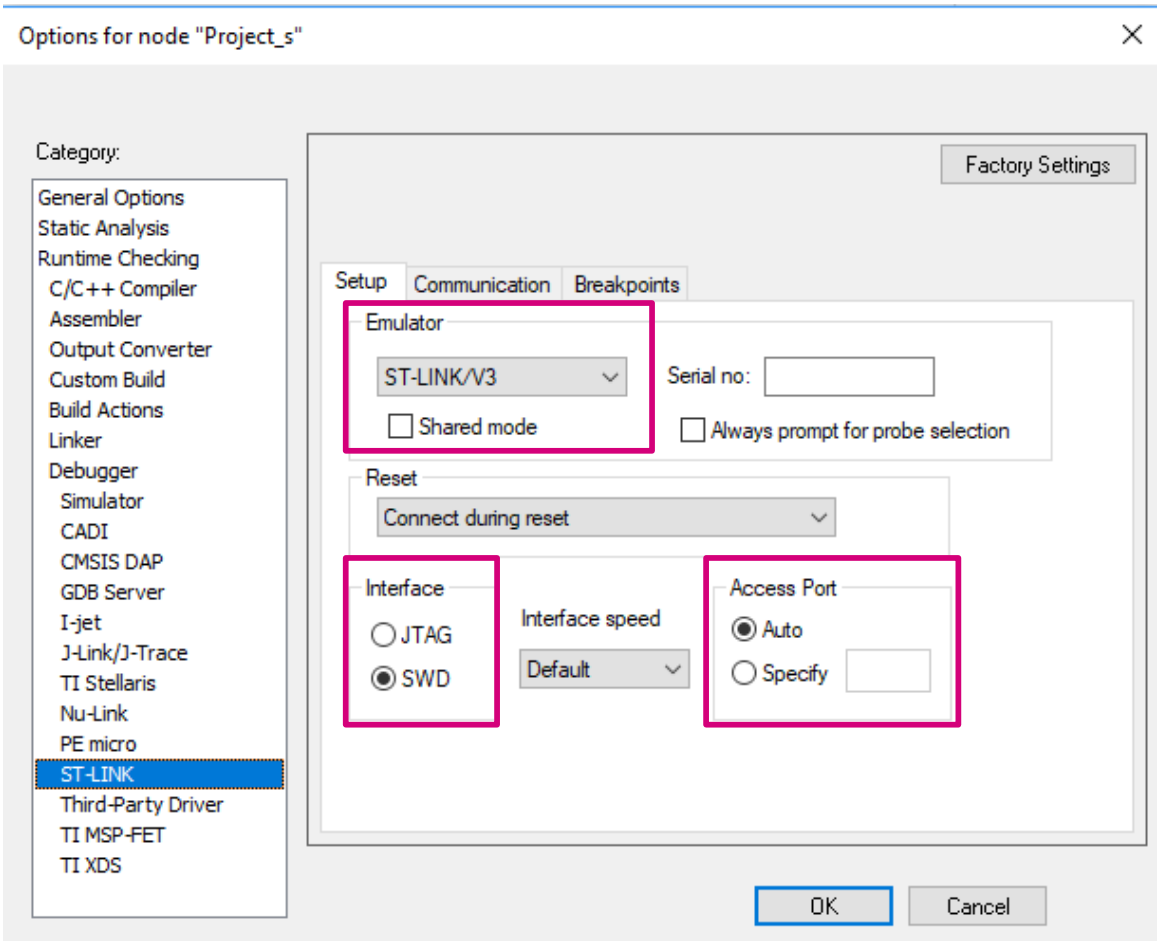
图 36. 选择非安全输出文件作为额外映像



调试信息会使调试器只下载调试信息，而不是完整的调试文件。

8. 从 Project / Options / ST-LINK “Setup”选项卡，参见图 37：
   - 选择“ST-LINK debugger”。
   - 选择复位类型：
     - 系统复位：复位内核和外设。
     - 内核复位：通过 VECTRESET 位复位内核；外设单元不受影响。
     - 软件复位：设置 PC 为程序入口地址。
     - 硬件复位：探针反转 JTAG 连接器上的 nSRST/nRESET 线来复位设备。这种复位通常也会复位外设单元。
     - 复位过程中的连接：在保持“Reset（复位）”激活的同时将 ST-LINK 连接到目标。复位被拉为低电平，并在连接目标时保持低电平。
   - 选择通信接口：
     - JTAG：使用 JTAG 接口。
     - SWD：使用 SWO 接口，该接口使用的引脚数量少于 JTAG。如果要使用串行线输出（SWO）通信信道，请选择 SWD。
   - 选择接入端口：
     - Auto（自动）：自动将接入端口 0 应用于 Cortex®-M33。
     - Manually（手动）：指定要使用的接入端口。

图 37. 项目设置



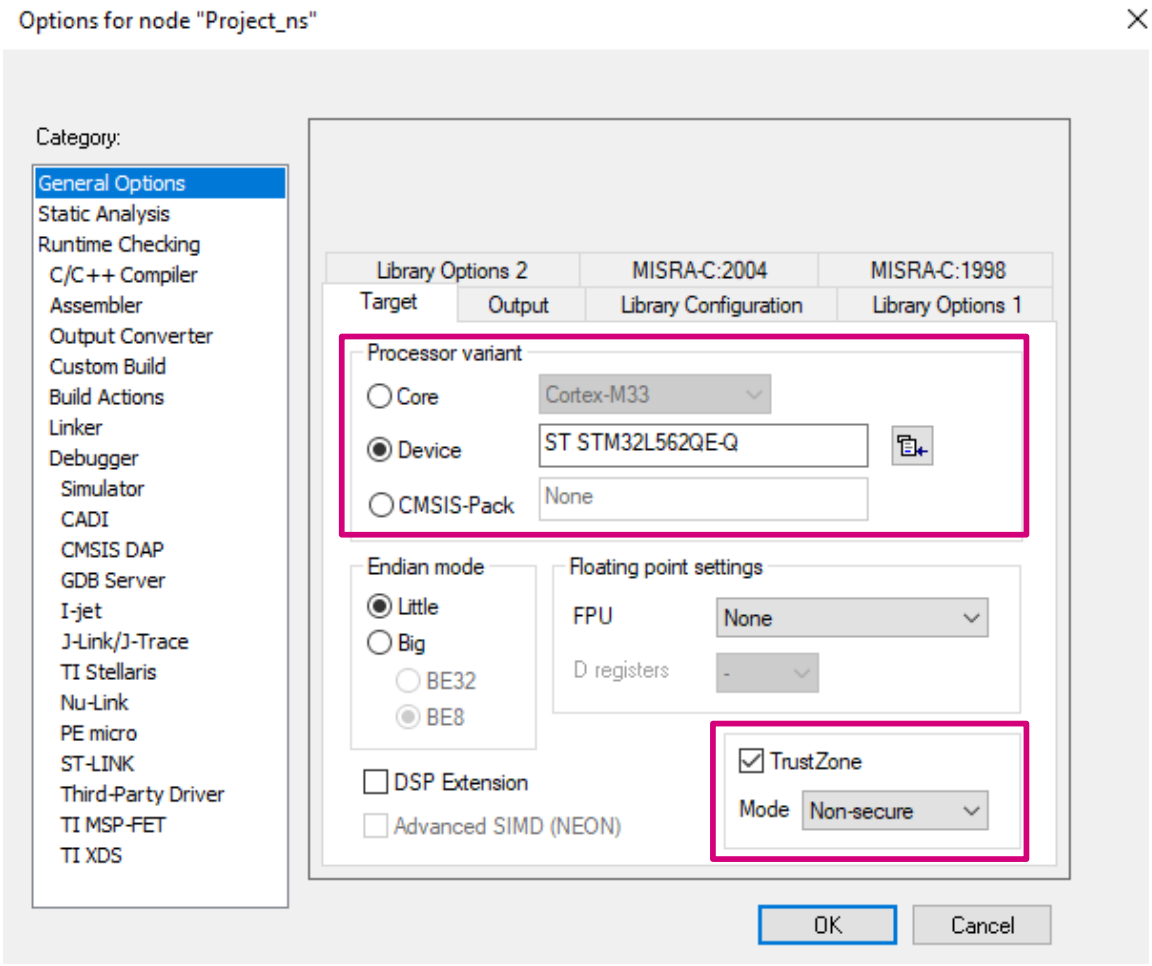
10.2 非安全项目设置

将 project_s-STM32L562E-DK_Templates_TrustZone 设为活动项目

1. 选择 Project-s / Options/ General Options（项目/选项/一般选项），从而打开配置窗口。在“目标”选项卡中，从处理器部分选择正确的器件（参见图 38）。

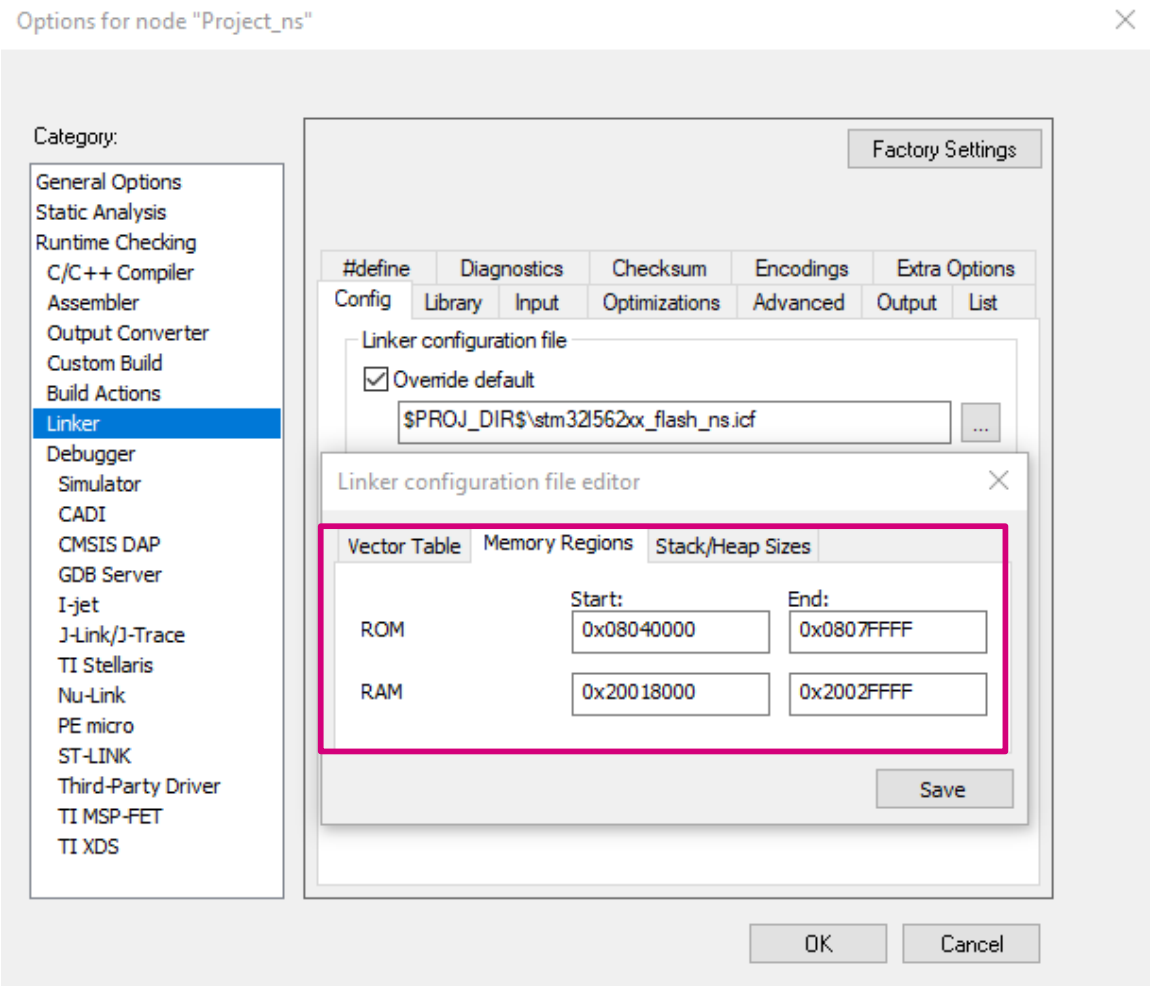
   从 TrustZone®部分，确保选择“非安全”模式，并且勾选 TrustZone®复选框。

图 38. 项目设置：一般选项



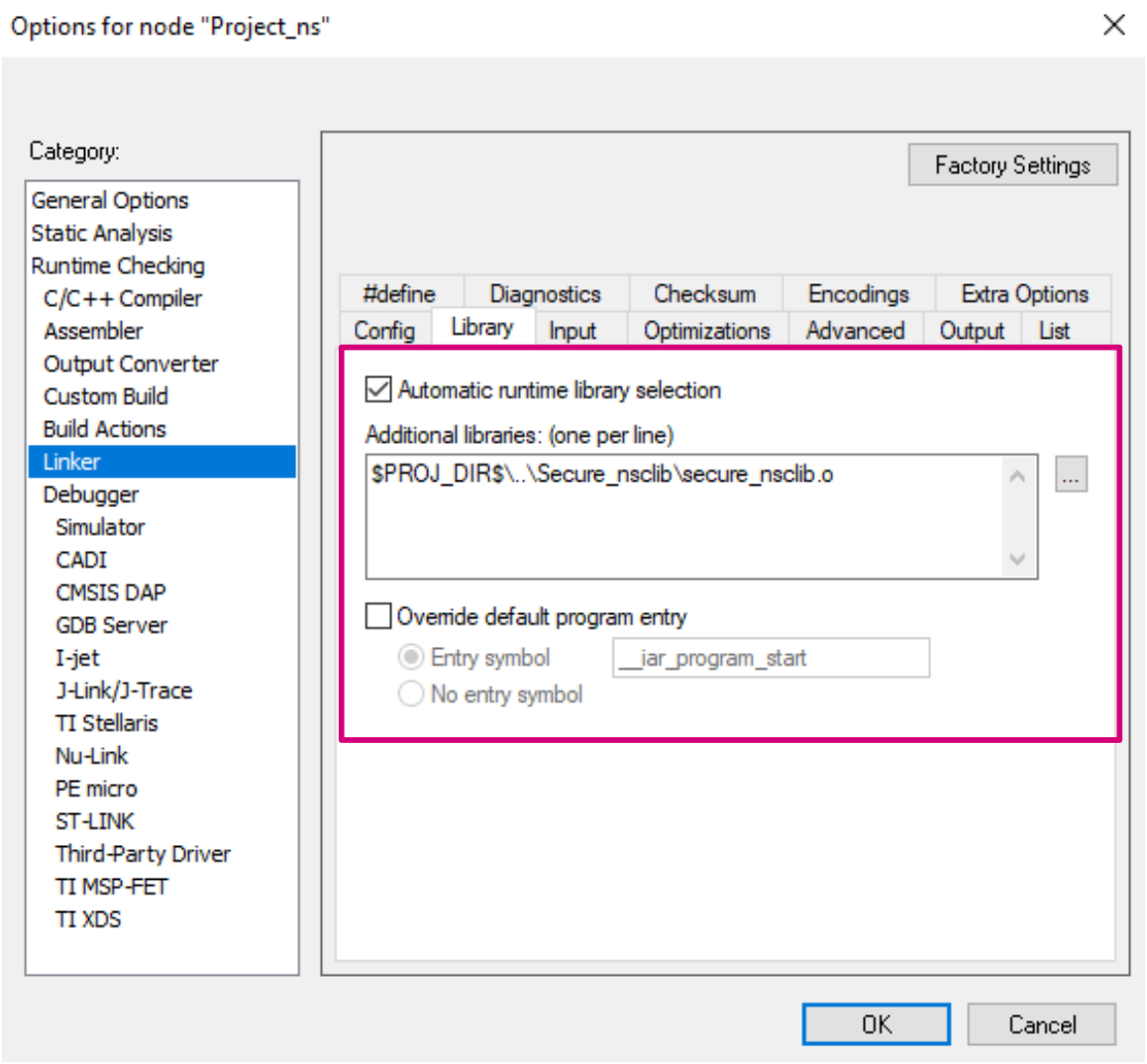
2. 从 Project-s / Options / Linker / Linker configuration（项目/选项/链接器/链接器配置）文件部分（参见图 39）：
   – 单击“编辑”以显示链接器配置文件编辑器。
   – 检查链接器配置文件，以确保应用程序已链接到正确的地址：
     ◦ 启动地址 0：位于 0x08040000 的闪存（非安全闪存）
     ◦ 启动地址 1：位于 0x20018000 的 SRAM（非安全 SRAM）。

图 39. 项目链接器配置



3. 从“库”中的 Project-s / Options / Linker（项目/选项/链接器）（参见图 40）。
添加从安全项目导入的库。此文件在链接时自动包含在非安全项目中。它允许非安全部分调用安全部分的函数。

图 40. 链接器库设置



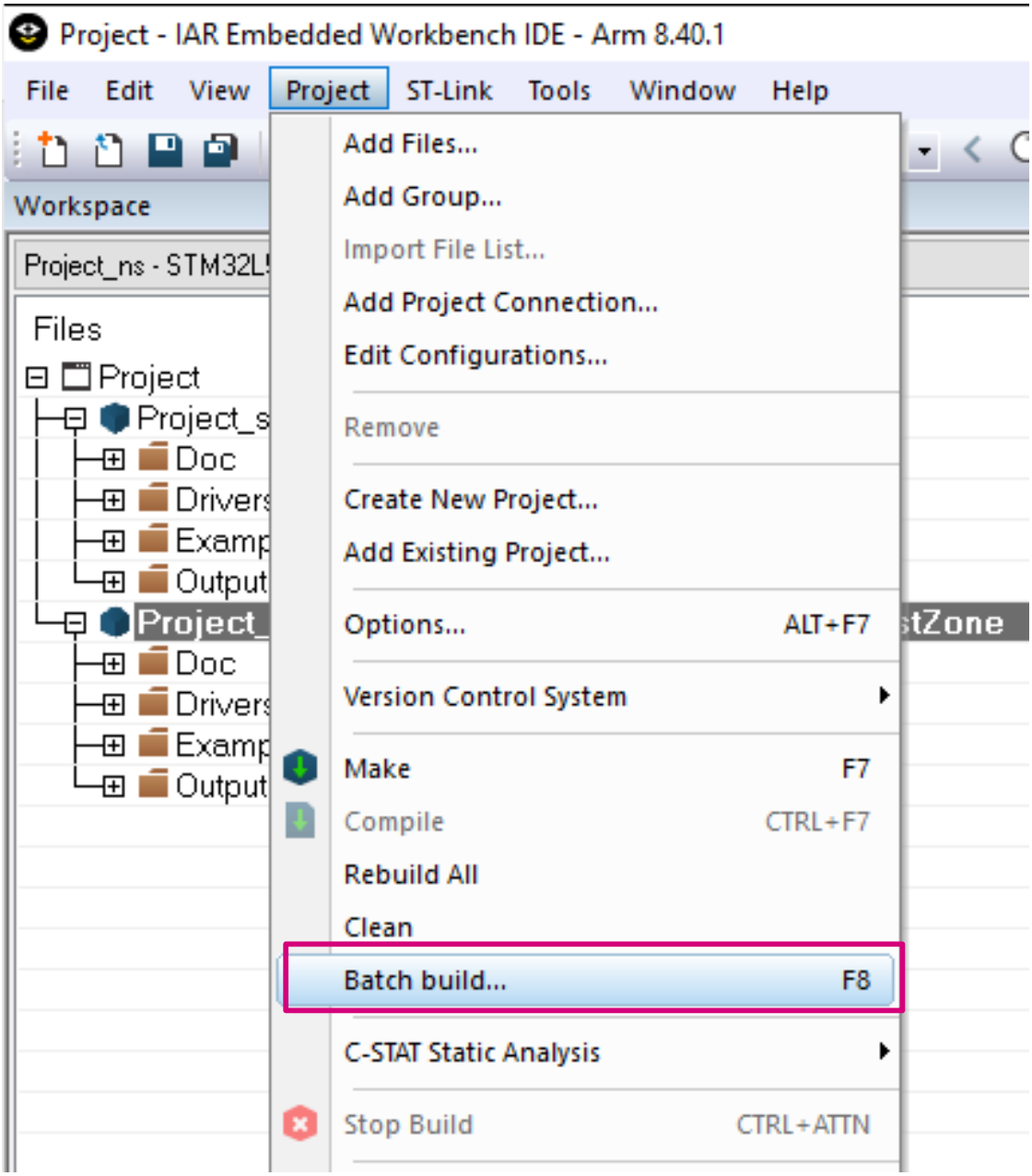
4. 其他配置类似于安全项目。

10.3 编译项目

两个项目都准备好进行编译。

1. 选择 Project / Batch Build（项目 / 批编译）或菜单栏中可用的图标（参见图 41）。

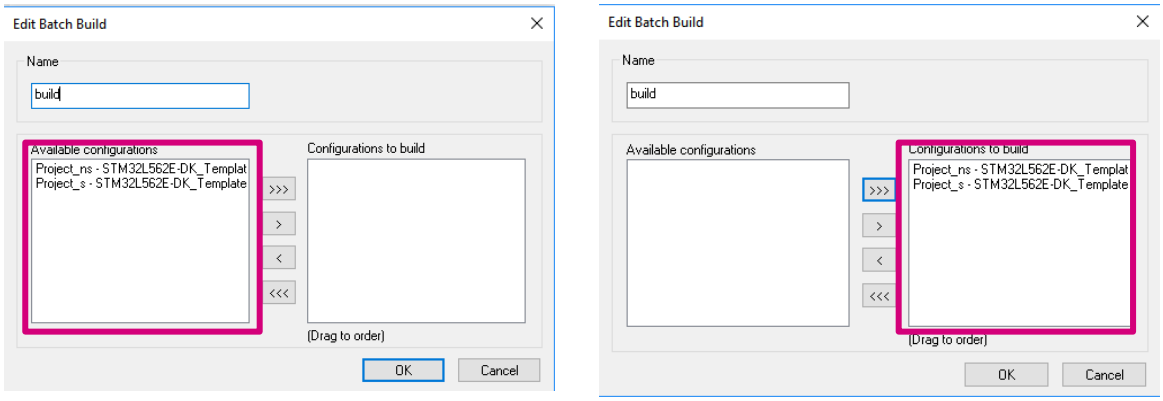
图 41. 项目批编译



2. 同时添加要编译的两个配置（参见图 42）。

提示 *必须首先编译安全项目，以便为非安全项目创建导入库。为了在编译非安全项目之前编译安全项目，它必须在编译顺序中处于首位，如下所示。*

图 42. 项目批编译顺序

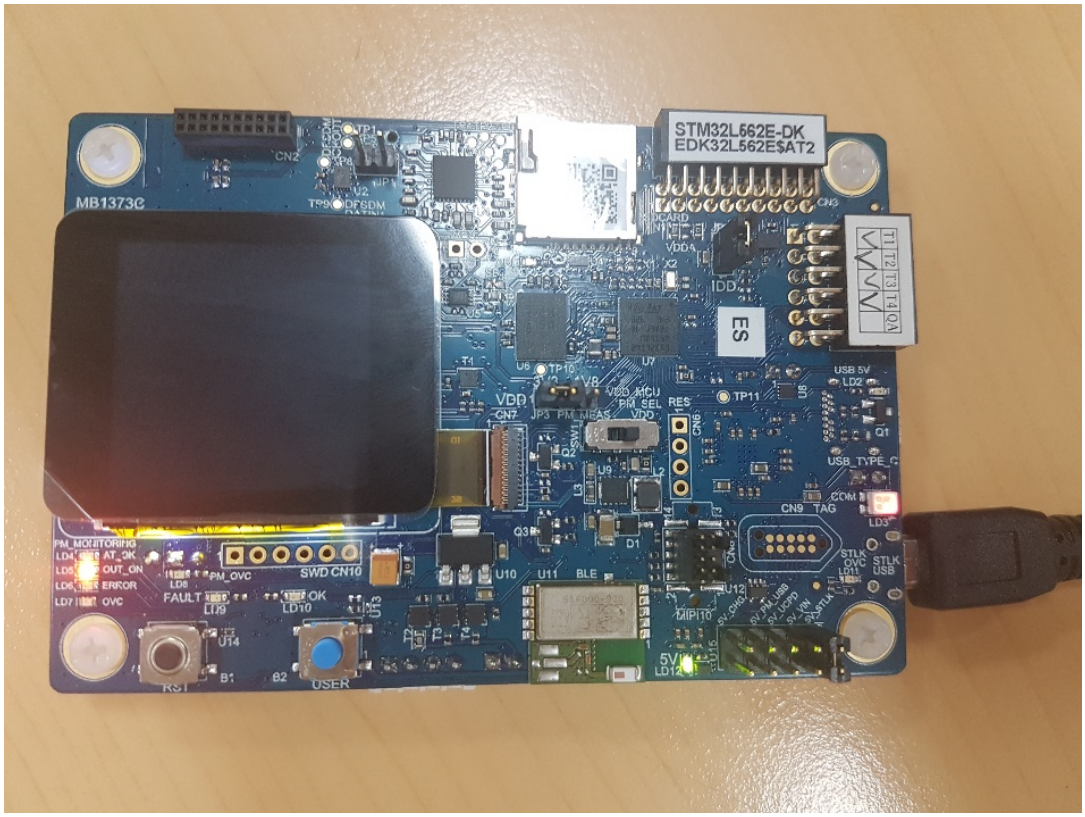


10.4 从安全代码执行到非安全代码

为了执行任意代码，它必须下载到板件，过程如下：

1. 下载项目之前，按如下方式连接到 STM32L562E-DK 探索板（参见图 43）：
   – 将 USB 电缆插入到探索板的 CN17 ST-LINK USB 连接器，从而将 ST-LINKV3 编程和调试工具连接到探索板。
   – 当 ST-LINKV3 连接后，LD3 亮起为红色。

图 43. **STM32L562E-DK 探索板处于连接状态**

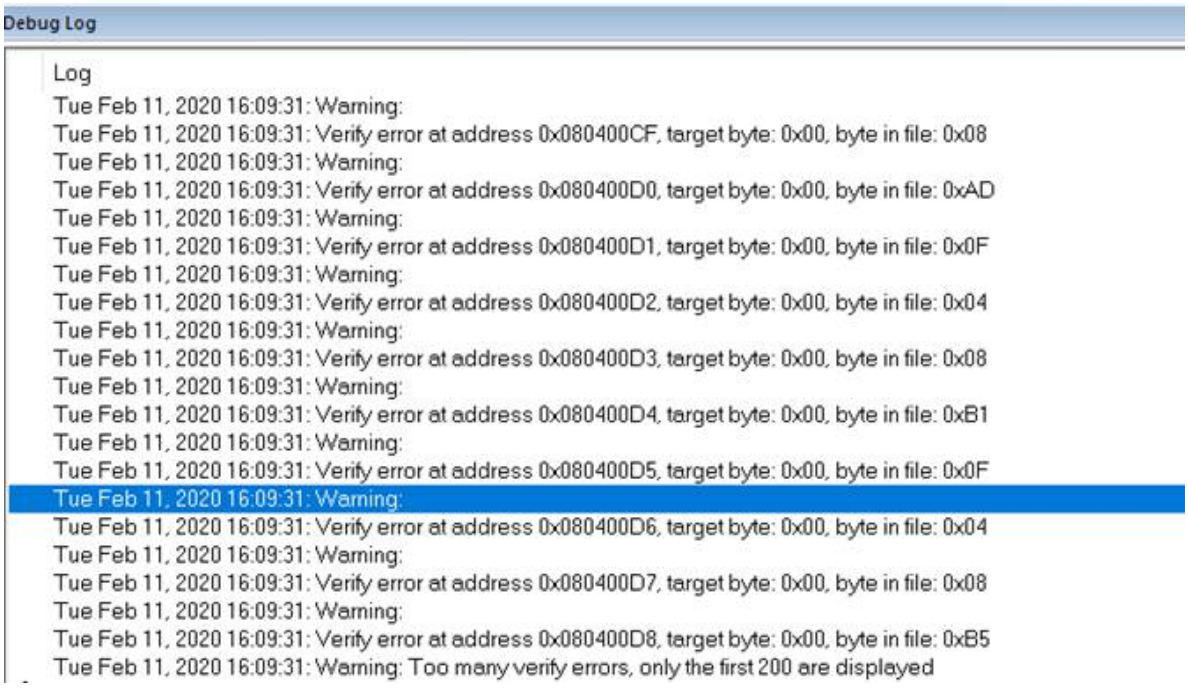


2. 选择 Project_ns 项目作为活动项目，然后加载非安全二进制代码。
单击工具栏中的下载和调试按钮来启动调试会话，以便对闪存进行编程并开始调试（参见图 44）。

图 44. 下载和调试启动按钮

提示 *当尝试加载非安全应用程序时，将显示以下警告信息。*

图 45. 加载错误警告信息示例的非安全应用程序



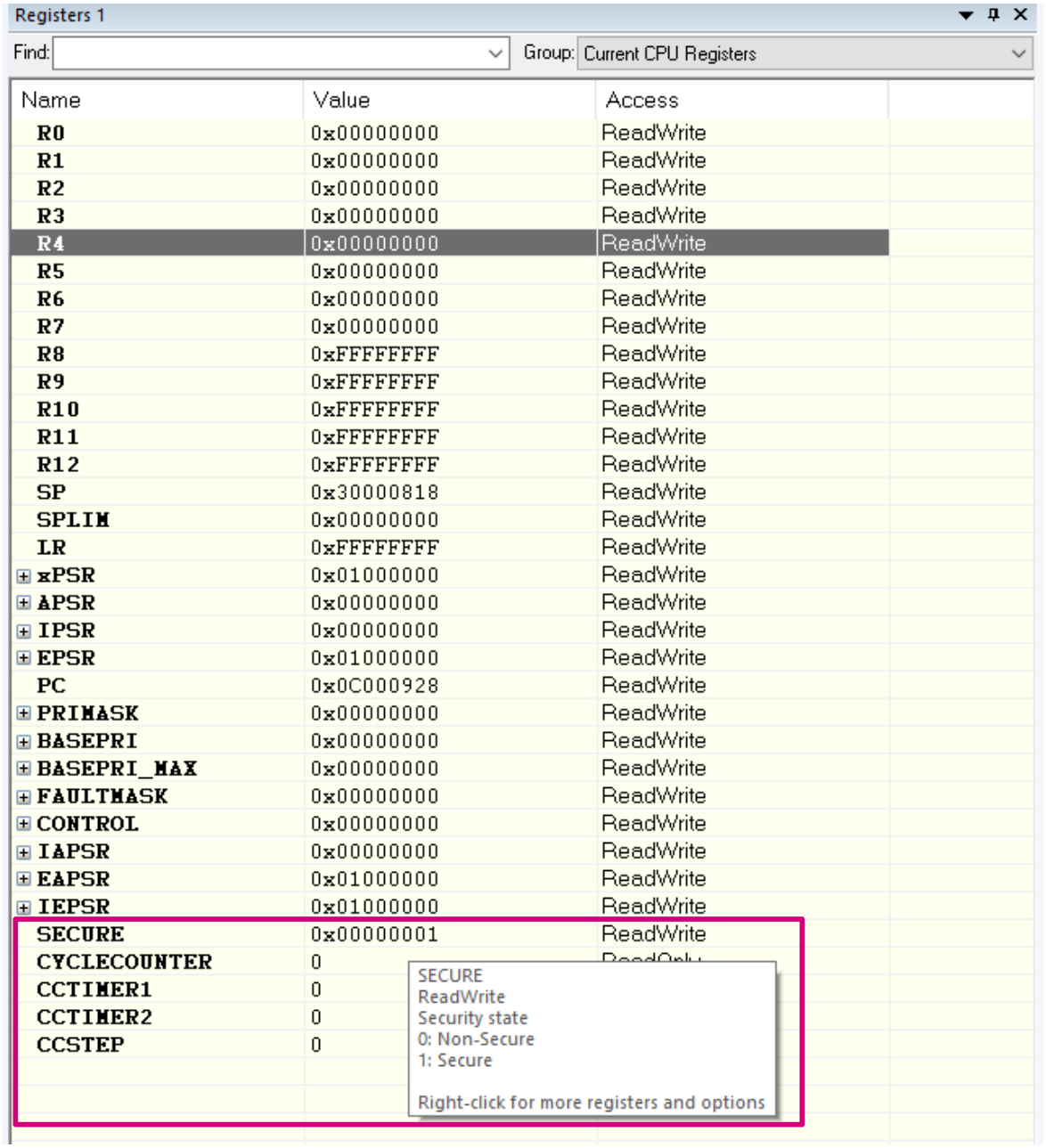
这是正常现象，因为在验证阶段，调试器会尝试回读加载的内容，并将其与编译的二进制文件进行比较。
在 SAU配置之前，调试器在非安全区域（@ 0x08040000非安全闪存）生成安全传输。此访问被禁止，并且内容读取为零。

3. 选择 Project_s 项目作为活动项目，然后加载非安全二进制文件，然后启动调试会话。

提示 *系统总是首先在安全代码（main.c）中启动，然后安全应用程序启动非安全应用程序*

4. 安全状态由 CPU 寄存器下的安全寄存器提供（参见图 46）。

图 46. 安全寄存器位置



0 = 非安全

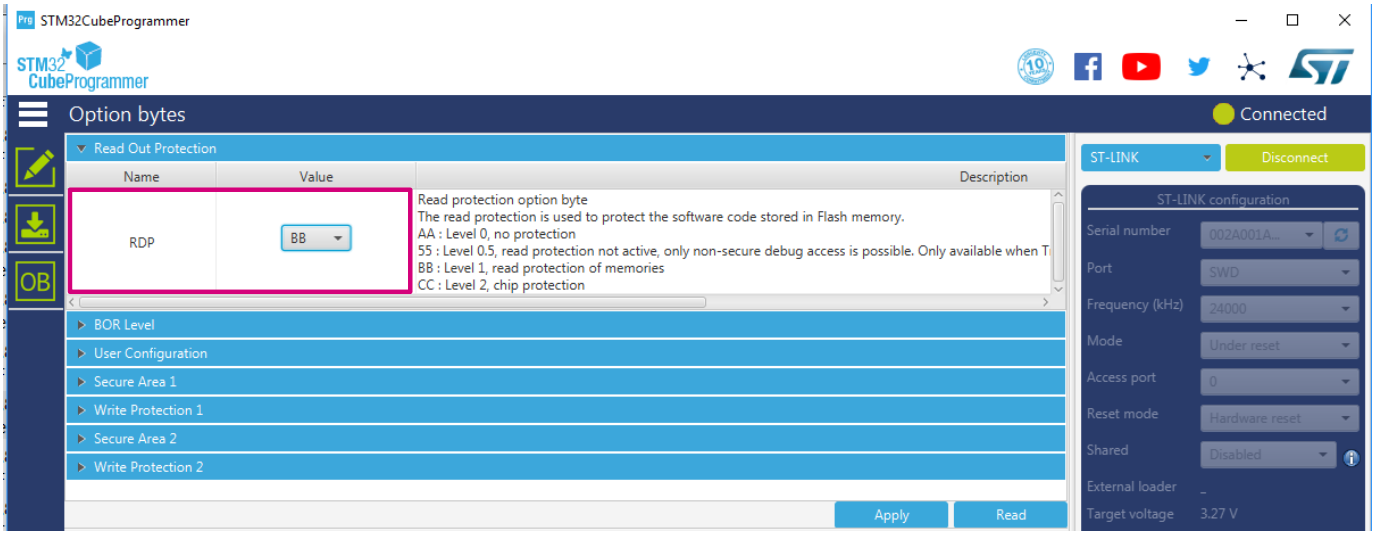
1 = 安全

10.5 当 RDP 设为 0.5 时，与 STM32L552ZE-Q 的连接问题

EWARM 能够连接到器件并调试非安全应用程序。要连接到 STM32L552ZE-Q，步骤如下：

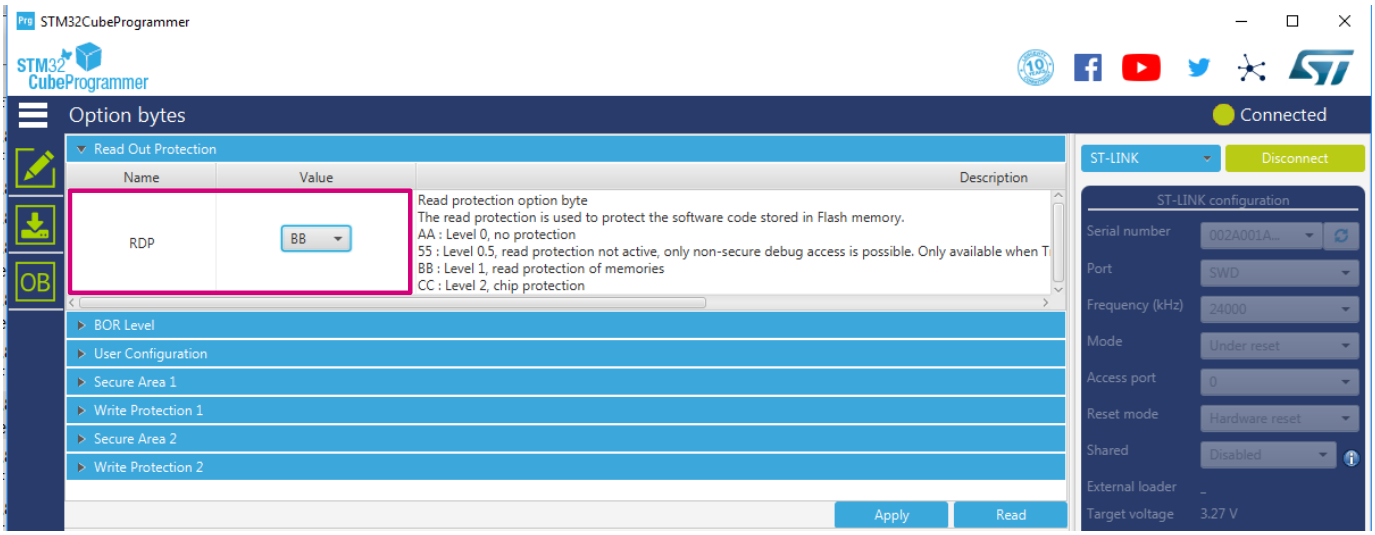
1. 设置选项字节，如图 47 中所示：
   - TZEN = 1
   - DBANK = 1
   - SECWM2_STRT = 0x1
   - SECWM1_PEND = 0x0.

图 47. 使用 STM32CubeProgrammer v2.2.0 配置选项字节



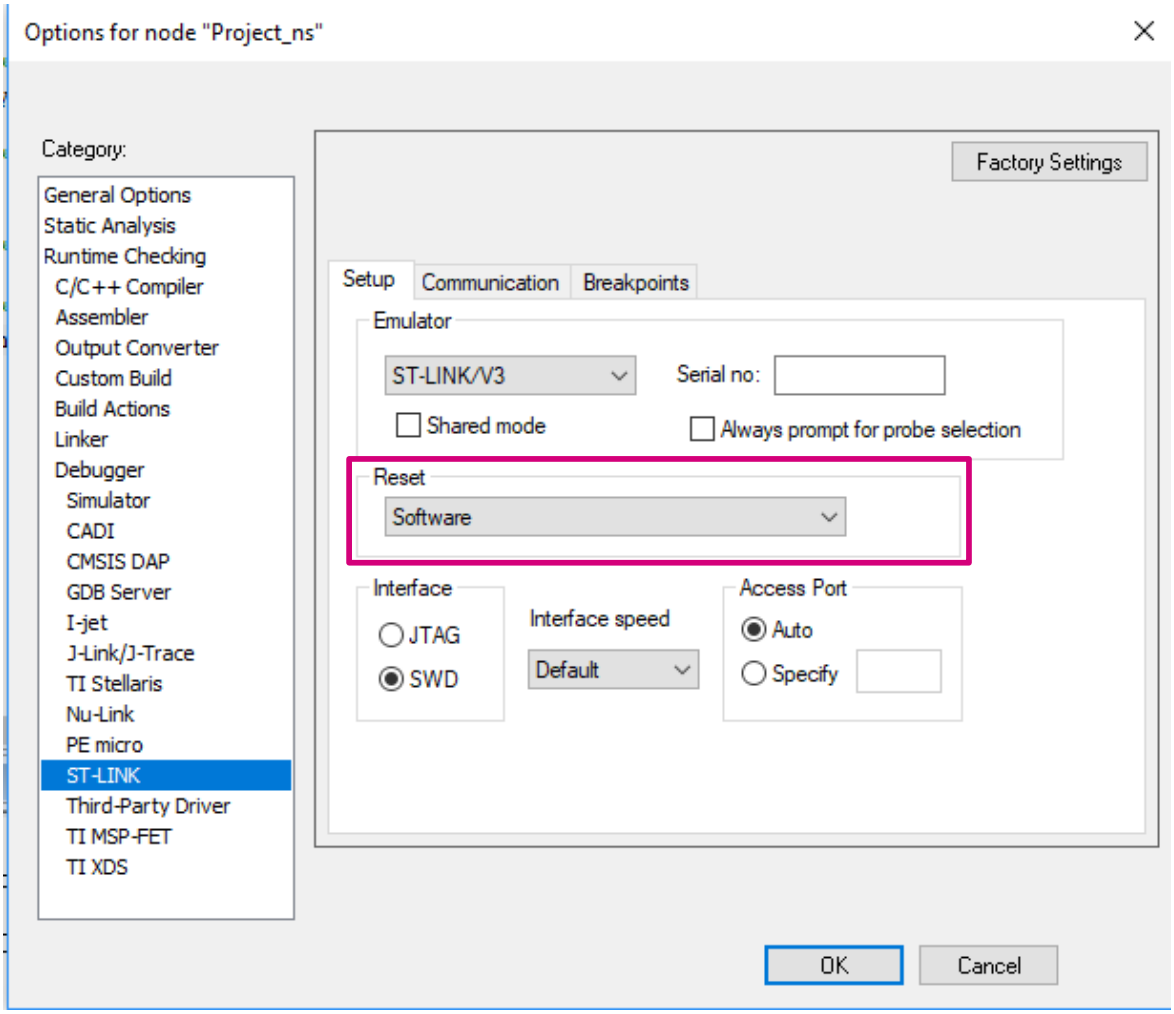
2. 加载非安全二进制（位于 0x08040000），然后加载安全二进制（位于 0x0C000000），如上节所述。
3. 使用 STM32CubeProgrammer 设置 RDP=0x55，以减少对非安全的调试（参见图 48）。

图 48. RDP=0.5



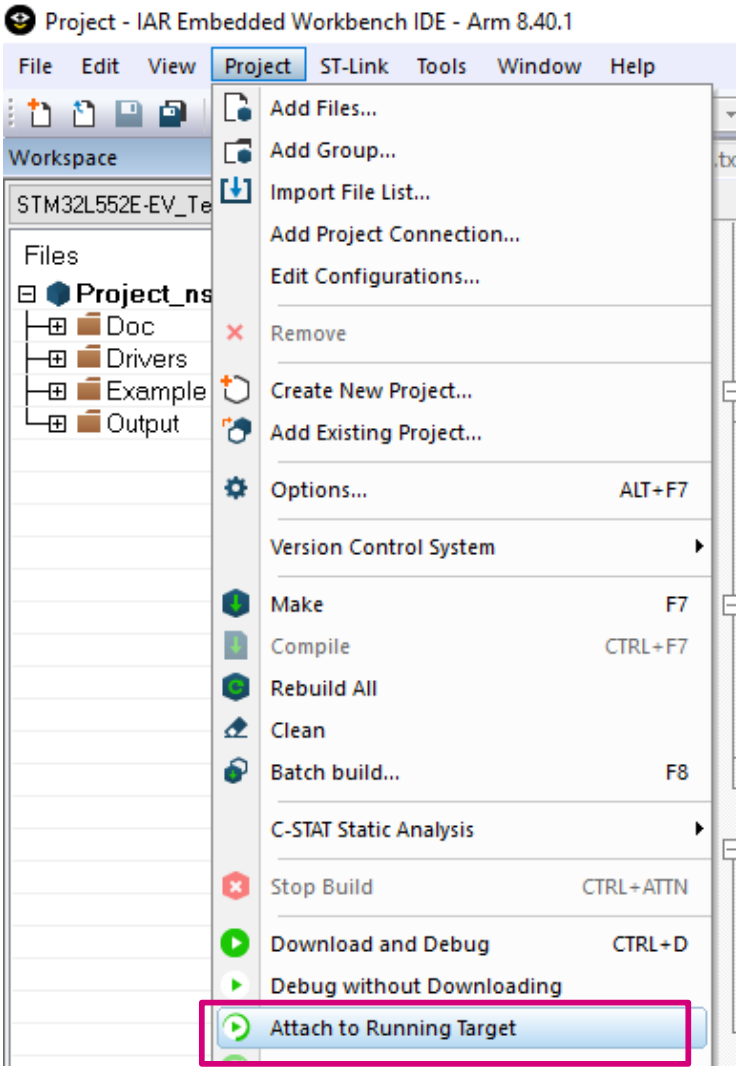
4. 将复位模式改为软件复位：“设置”选项卡中的 Project options / ST-LINK ，从“Setup”字段选择“Software”，如图 49 中所示。

图 49. 复位模式选择



5. 在热插拔模式下连接器件：Project / Attach to the Running Target（参见图 50）。

图 50. 附加于运行中的目标选项



提示　　IDE 不支持在 RDP level 0.5 下进行非安全 flash 再编程，只有 STM32Cubeprogrammer 允许。

11 将 CubeIDE 用于带 Trust Zone® 的 Cortex®-M33

本部分将在 *CubeIDE 中的 STM32 开发入门*（AN5394）中说明，后者可从 *www.st.com* 上获取。

版本历史

表 1. 文档版本历史

日期	版本	变更
2020 年 2 月 21 日	1	初始版本。

目录

图一览