life.augmented

STM32 生态系统 第二十一期

信息安全 . Information Security

12. X-Cube-SBSFU/STM32G0

2020.03

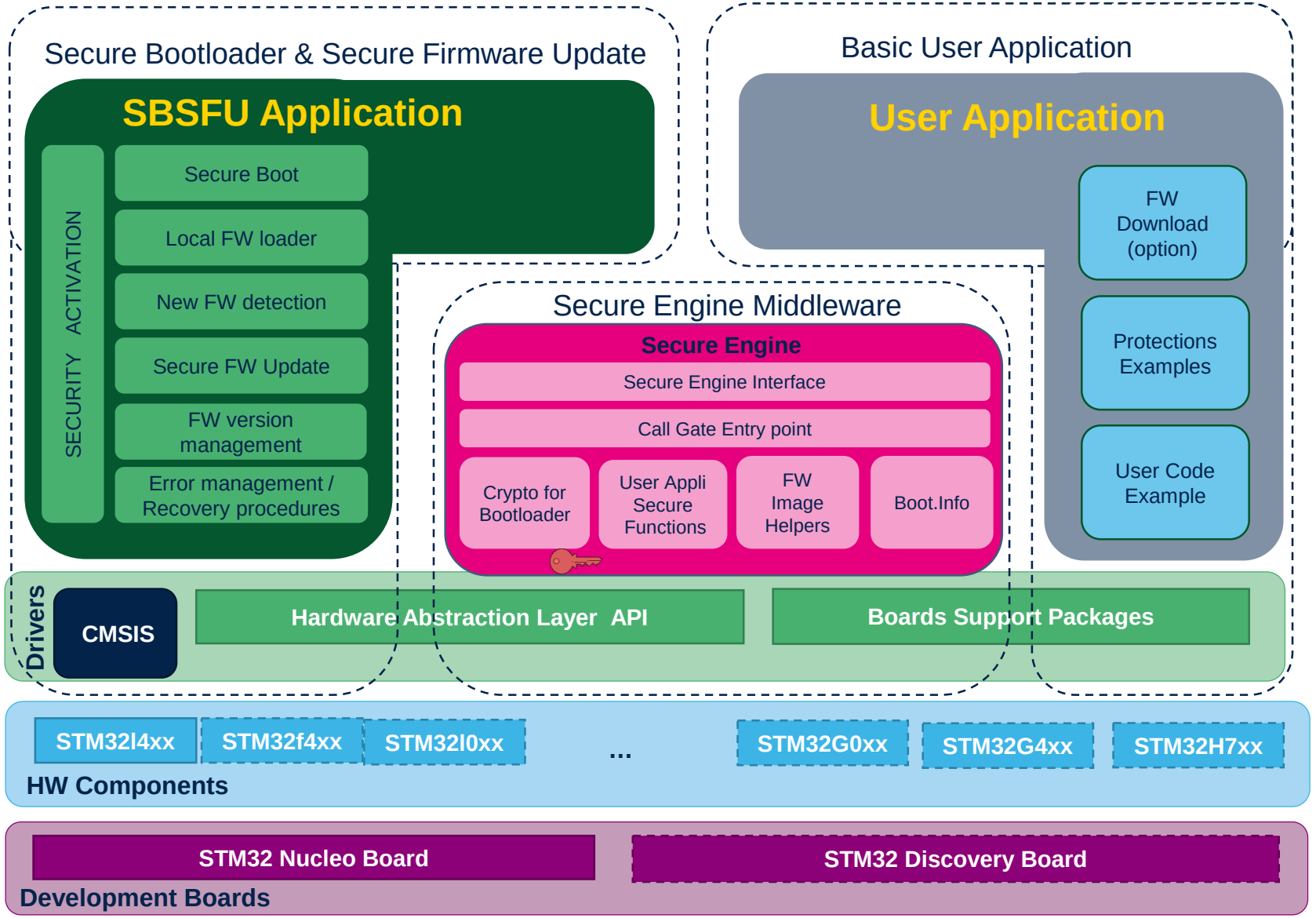
• 深入了解：X-Cube-SBSFU软件包

X-Cube-SBSFU 深入了解

- X-Cube-SBSFU软件包的定位
  - ST提供给客户的，用于解决“安全启动”和“安全固件更新的”的，开源、免费的参考实现
  - 客户可以基于此，定制开发自己的实现，以满足实际应用的安全需求
- X-Cube-SBSFU软件包概览
  - 支持的芯片系列：
    - F4/F7/L1, G0/G4/H7,
    - WB, L4/L0,
  - 相关文档：
    - UM2262 X-Cube-SBSFU软件包入门
    - AN5056 X-Cube-SBSFU软件包集成指南
- 例程分支
  - 用户应用代码单image → I image
  - 用户应用代码双image → 2 images
  - 使用开源密码库mbedTLS → 2 images OSC
  - 使用STSAFE存储证书 → 2 images STSAFE
  - 支持密钥管理系统 → 2 images KMS

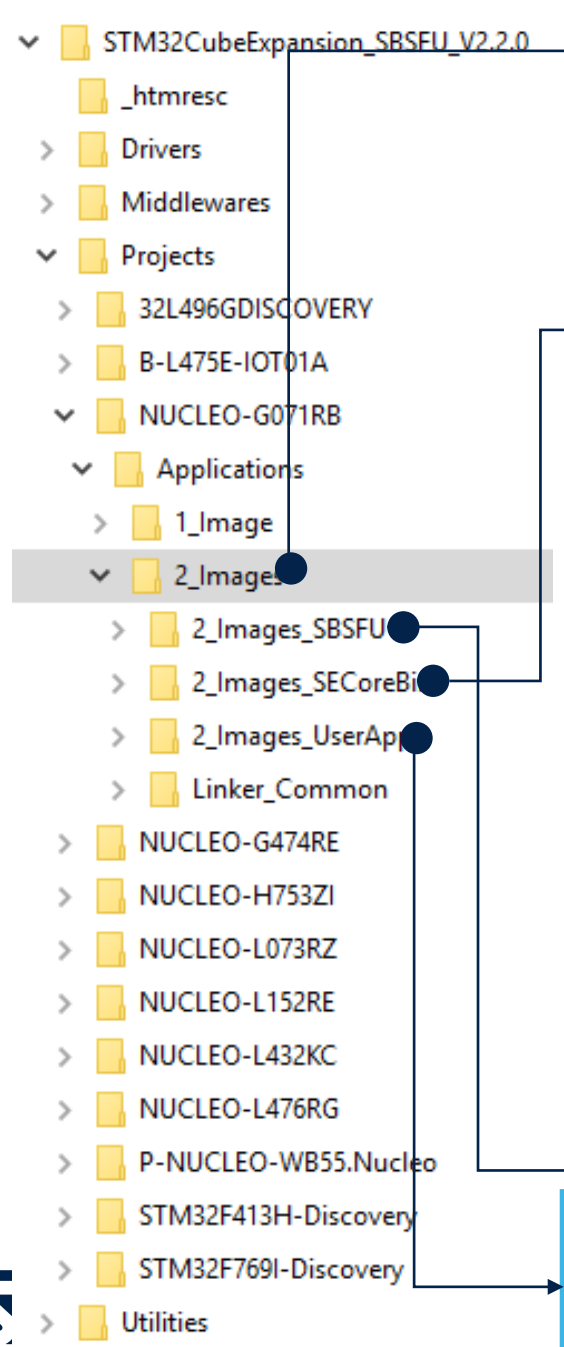
life.augmented

X-Cube-SBSFU 架构概览



life.augmented

Readme



readme.txt：三个项目的编译顺序，各自的输入、输出

readme.txt：

- 在头文件定义用户采用的加密方案（三选一）
- 根据prebuild.bat生成se_keys.c，将对称秘钥、ECDSA公钥hard code到代码中
- 生成postbuild.bat，用于后续UserApp项目的编译、链接

se_crypto_config.h

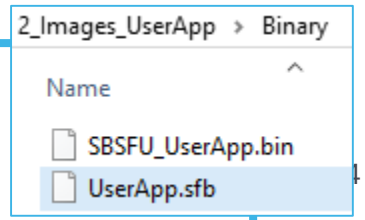
```
#define SECBOOT_CRYPTO_SCHEME SECBOOT_ECCDSA_WITH_AES128_CBC_SHA256 /*!< Selected Crypto Scheme */

#define SECBOOT_ECCDSA_WITHOUT_ENCRYPT_SHA256        (1U) /*!< asymmetric crypto, no FW encryption          */
#define SECBOOT_ECCDSA_WITH_AES128_CBC_SHA256        (2U) /*!< asymmetric crypto with encrypted Firmware    */
#define SECBOOT_AES128_GCM_AES128_GCM_AES128_GCM     (3U) /*!< symmetric crypto                             */
```

readme.txt：

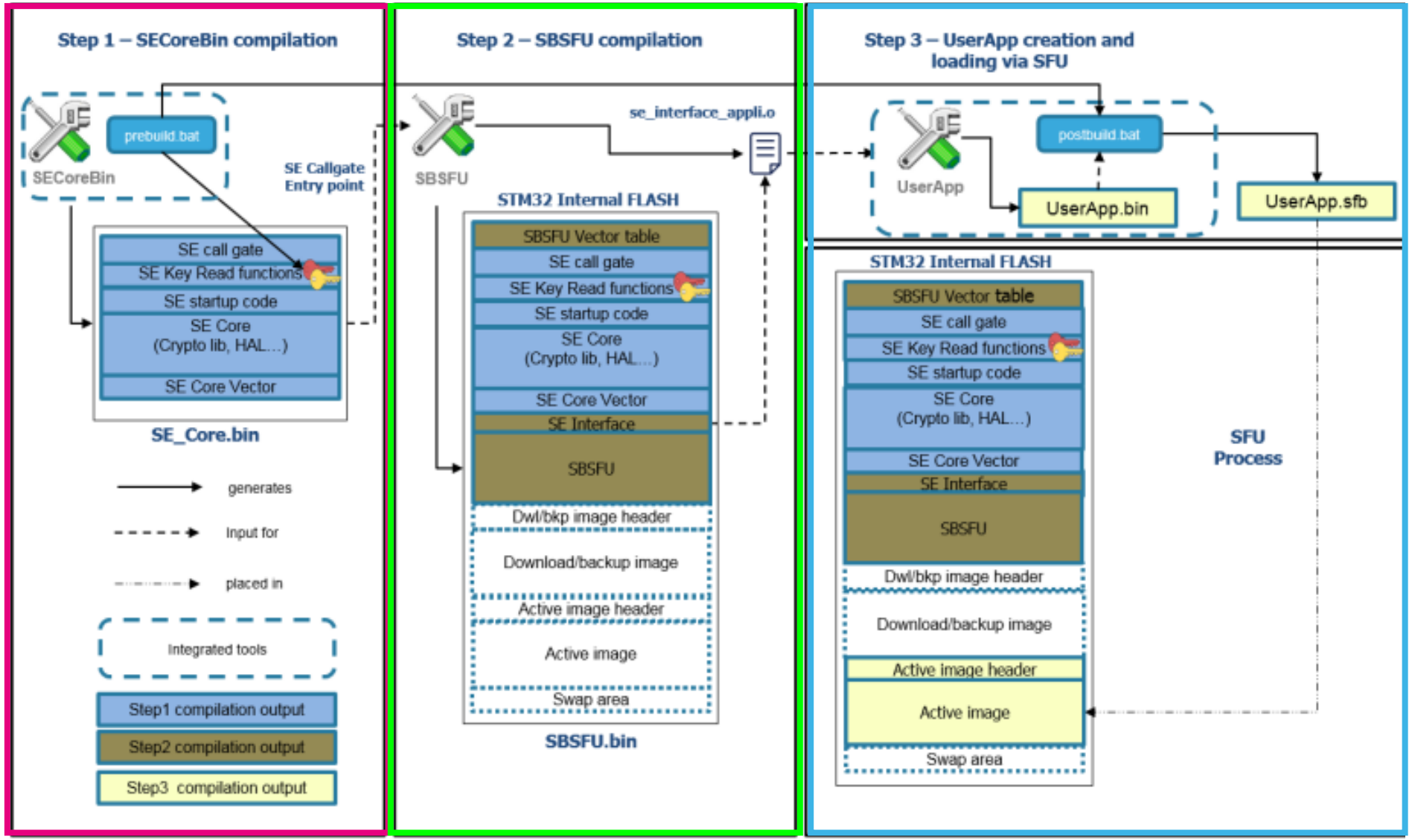
- 安全启动 & 安全固件升级
- 采用2image-2slot方案：下载新固件；固件更新可断点续更
- 下载前，确保芯片的状态（选项字节：RDP/PCROP/WRP/Boot_lock/nBOOT_SEL…）
- 运行时，需要串口工具做用户交互

readme.txt：

- 下载新固件、测试保护机制
- UserApp.bin，通过postbuild.bat生成UserApp.sfu，SBSFU_UserApp.bin



X-Cube-SBSFU 项目间的耦合



工程一（secorebin）的输入输出

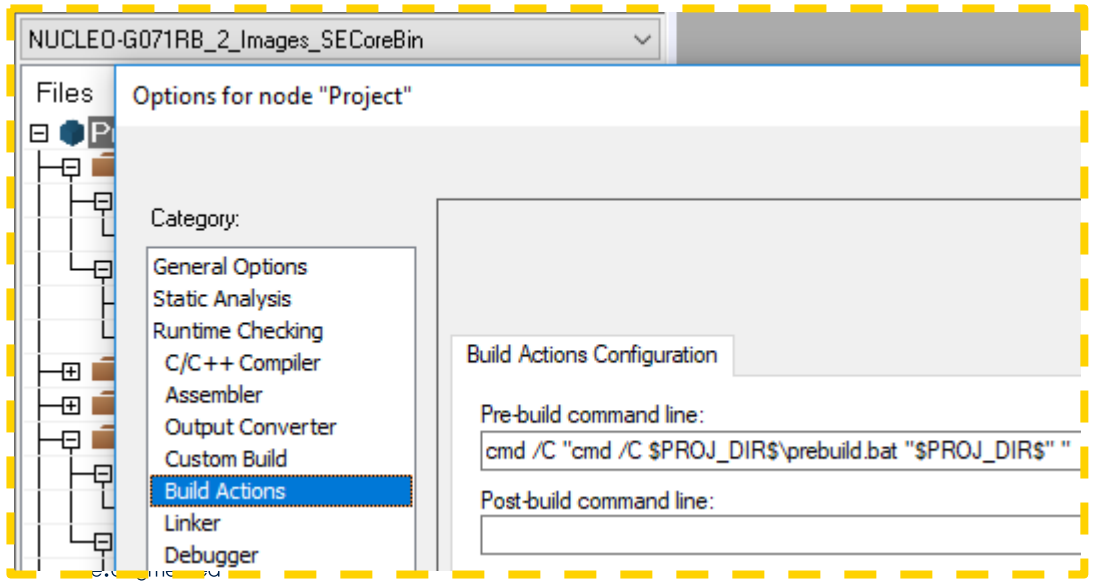
2_Images_SECoreBin\
Binary\ECCKEY.txt

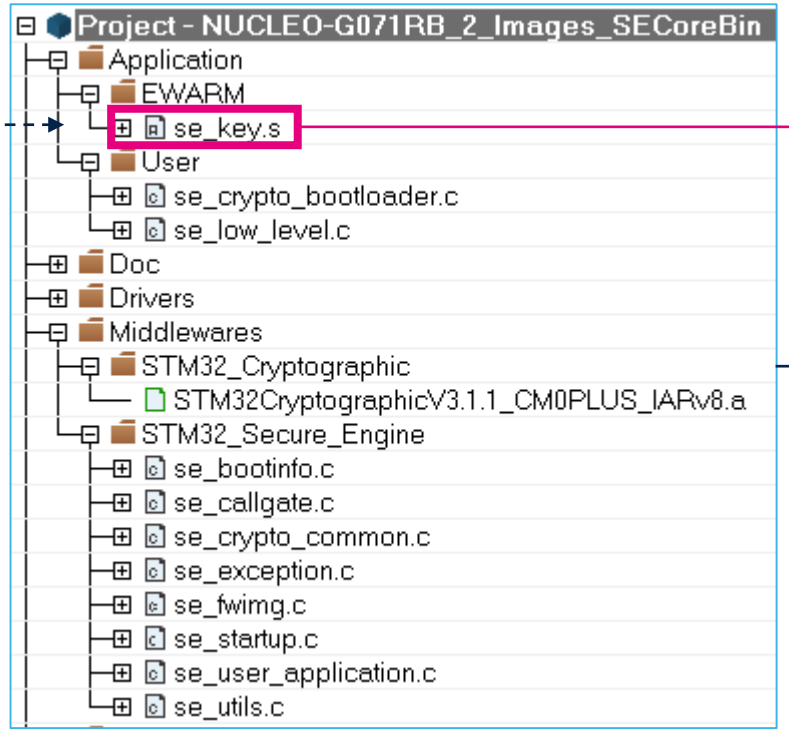
2_Images_SECoreBin\
Binary\OEM_KEY_COMPANY1_key_AES_CBC.bin

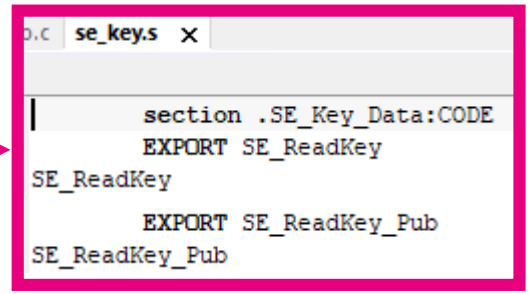
prebuild.bat

2_Images_SECoreBin\
EWARM\se_key.s

2_Images_SECoreBin\
EWARM\postbuild.bat



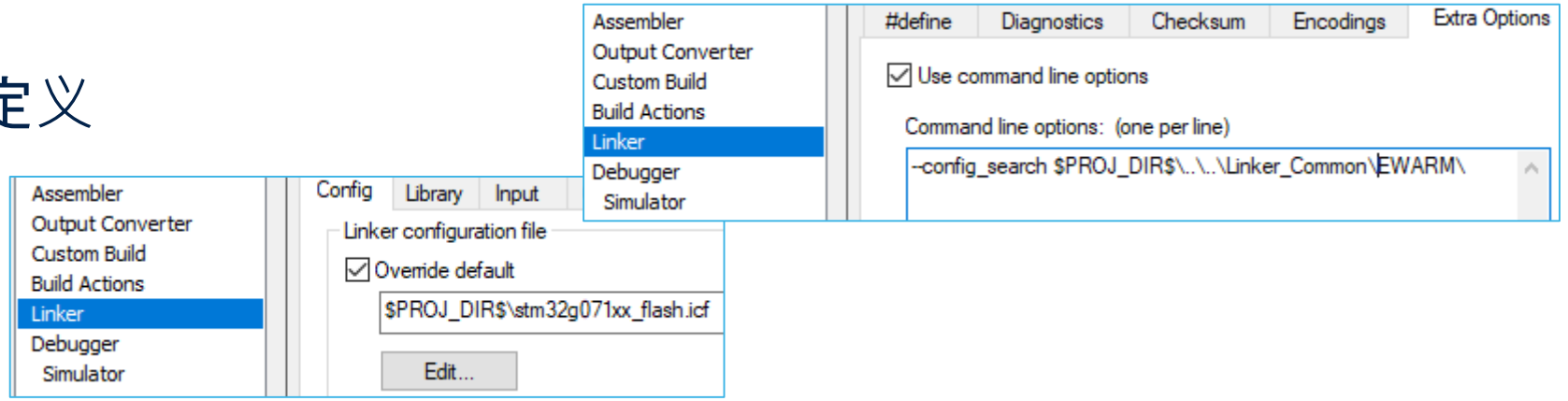




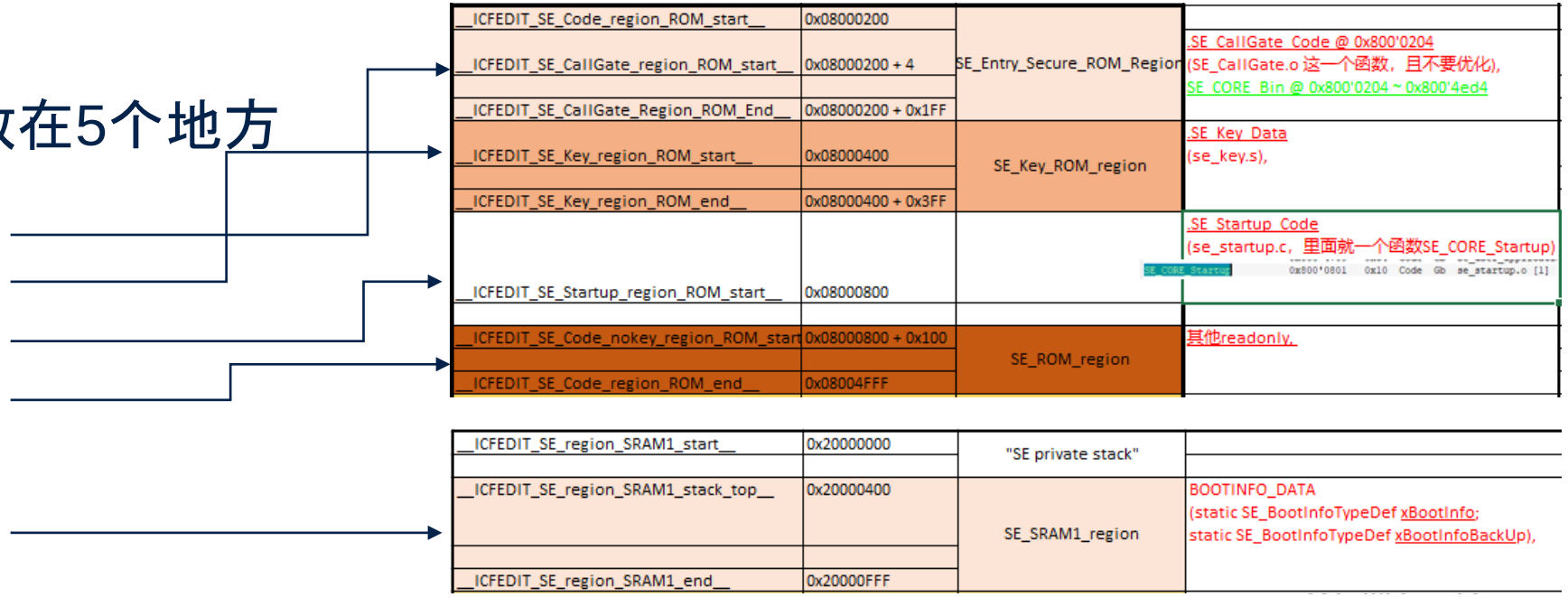
2_Images_SECoreBin\
EWARM\SE_Core.bin

工程一（SECorebin）的链接文件

- 根据三个linker文件的定义
  - stm32g071_flash.icf
  - mapping_sbsfu.icf
  - mapping_fwimg.icf



- 生成的data和code，放在5个地方
  - code中3个显式section
    - .SE_CallGate_Code
    - .SE_Key_Data
    - .SE_Startup_Code
  - code的其余部分
  - data @ BOOTINFO_DATA

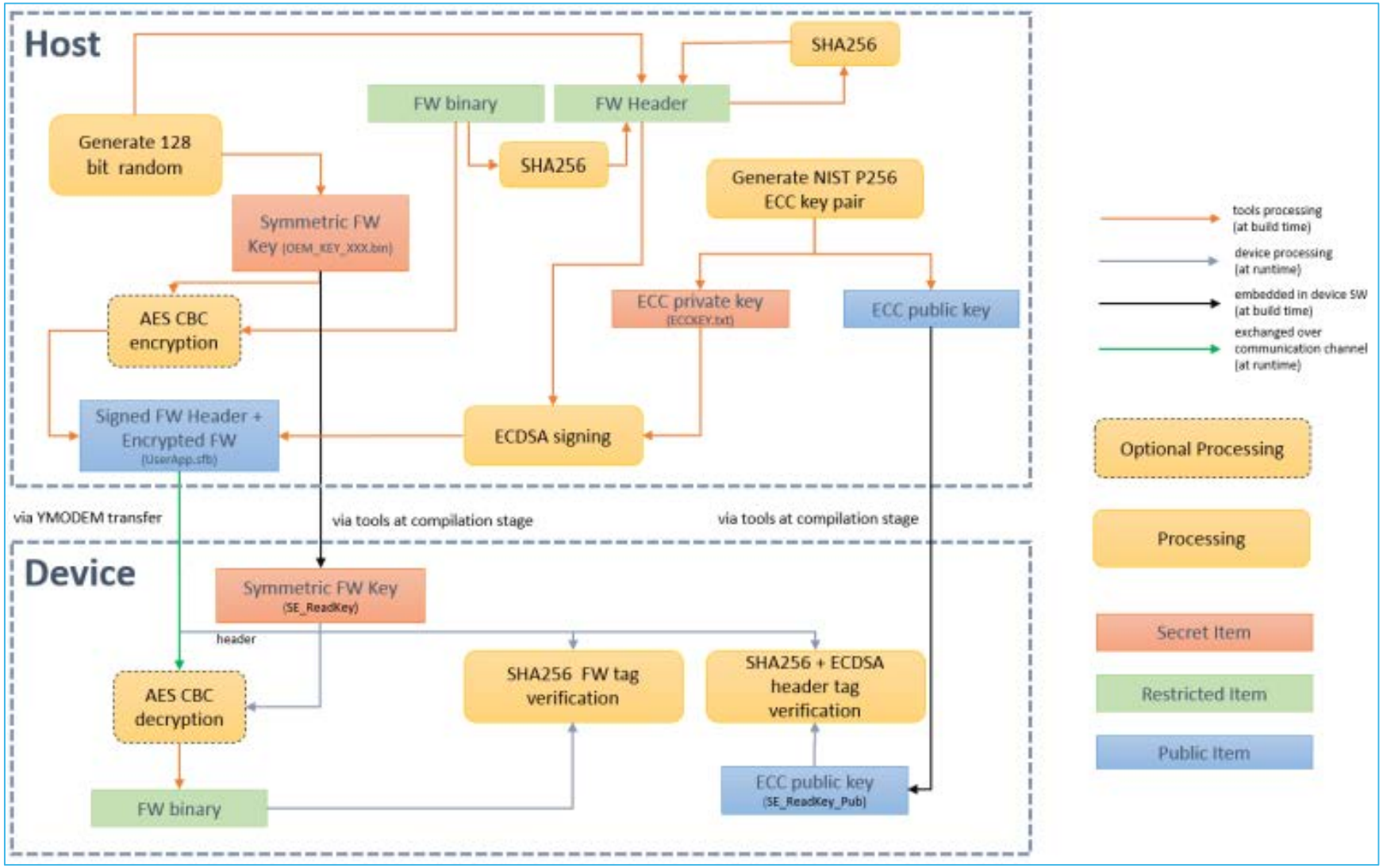


加解密方案

secboot_eccdsa_with_aes128_cbc_sha256

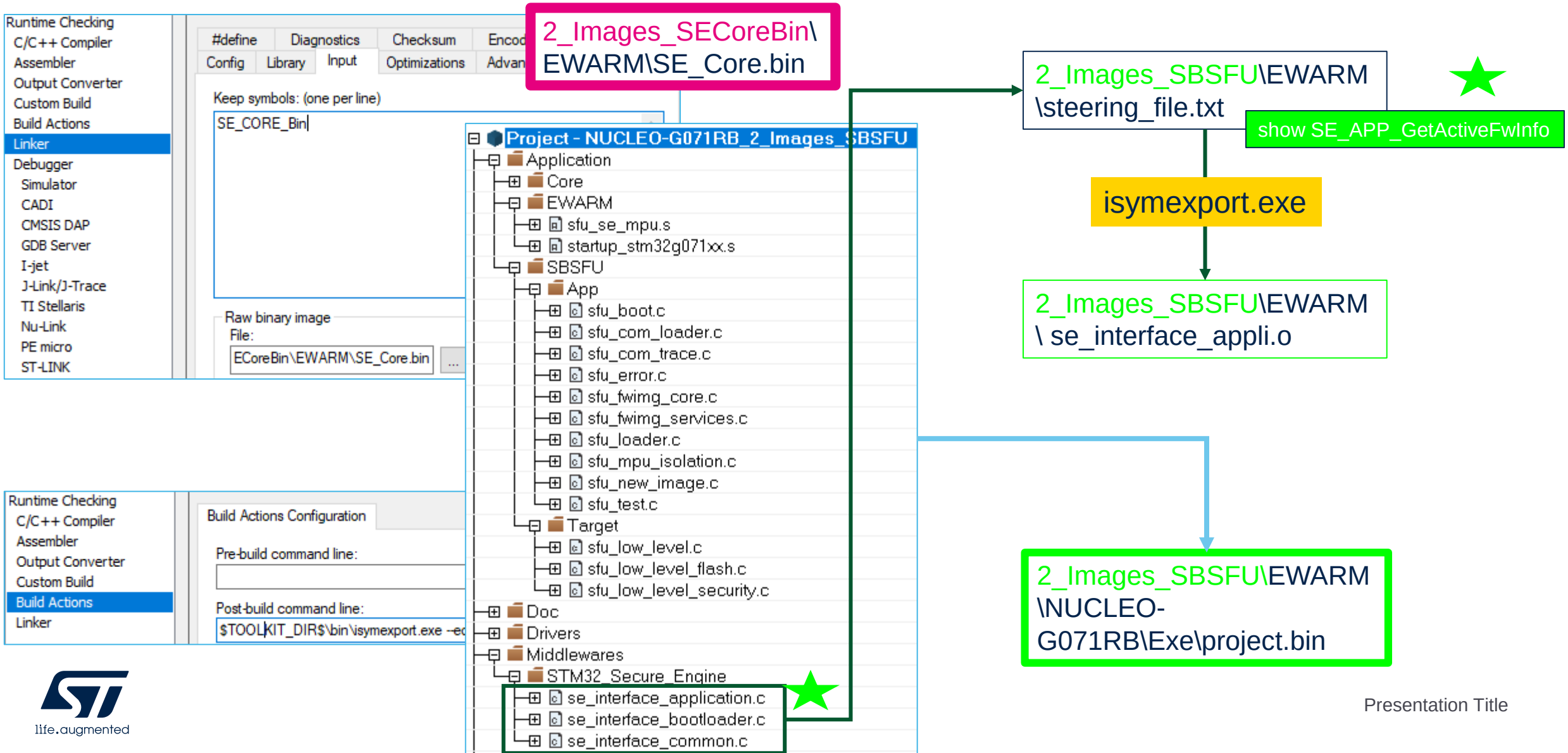
```
se_crypto_config.h
#define SECBOOT_CRYPTO_SCHEME SECBOOT_ECCDSA_WITH_AES128_CBC_SHA256

#define SECBOOT_ECCDSA_WITHOUT_ENCRYPT_SHA256        (1U)
#define SECBOOT_ECCDSA_WITH_AES128_CBC_SHA256        (2U)
#define SECBOOT_AES128_GCM_AES128_GCM_AES128_GCM     (3U)
```

- 加密方式
  - AES128_CBC
- 完整性算法
  - SHA256
- 认证方式
  - ECCDSA



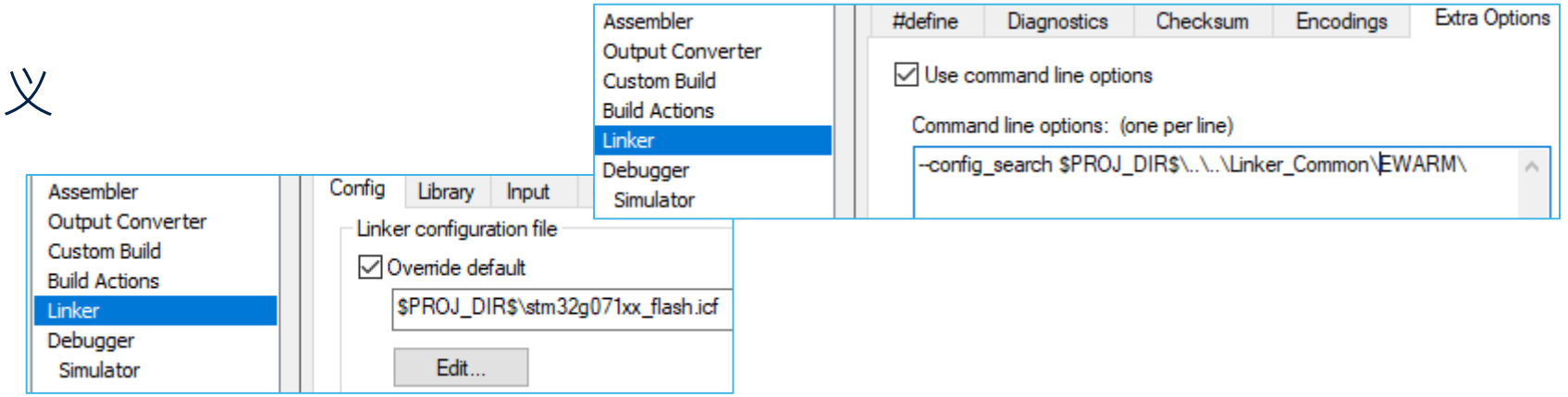
life.augmented

工程二（SBSFU）的输入输出



项目二（SBSFU）的链接文件

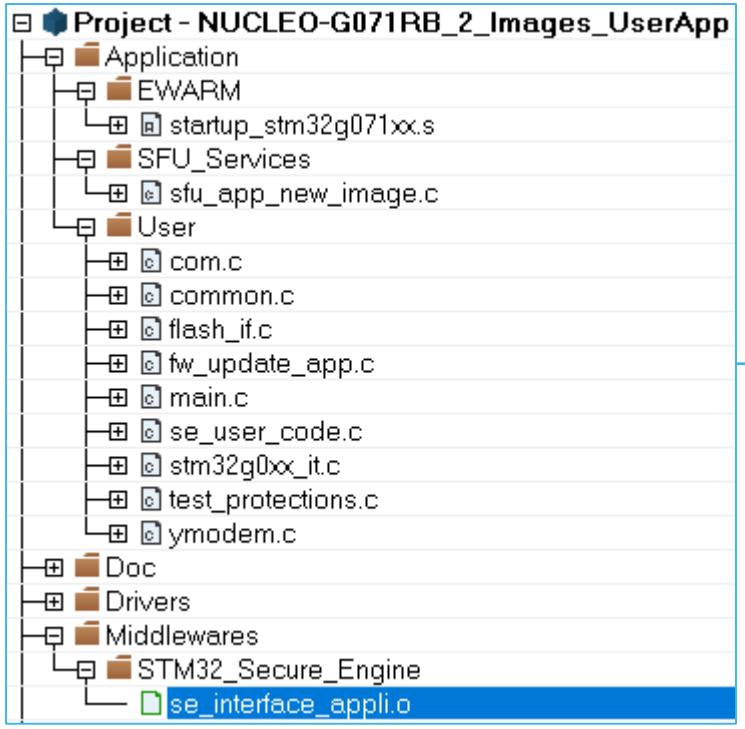
- 根据三个linker文件的定义
  - stm32g071_flash.icf
  - mapping_sbsfu.icf
  - mapping_fwimg.icf
- 生成的data和code，放在5个地方
  - code中3个显式section
    - .intvec
    - .SE_Core_Bin
    - .SE_IF_Code
  - code的其余部分
  - data @ CSTACK, HEAP



__ICFEDIT_SB_region_SRAM1_start__ __ICFEDIT_region_RAM_start__	0x20001000	SB_SRAM1_region	CSTACK, HEAP, CSTACK, HEAP, aes_block_padding // 16字节对齐
__ICFEDIT_SB_region_SRAM1_end__ __ICFEDIT_region_RAM_end__	0x20008FFF		

__ICFEDIT_intvec_start__	0x08000000		.intvec
__ICFEDIT_SE_Code_region_ROM_start__	0x08000200	SE_Entry_Secure_ROM_Region	.SE_CallGate_Code @ 0x800'0204 (SE_CallGate.o 这一个函数，且不要优化), SE_CORE_Bin @ 0x800'0204 ~ 0x800'4ed4
__ICFEDIT_SE_CallGate_region_ROM_start__	0x08000200 + 4		
__ICFEDIT_SE_CallGate_Region_ROM_End__	0x08000200 + 0x1FF		
__ICFEDIT_SE_Key_region_ROM_start__	0x08000400	SE_Key_ROM_region	.SE_Key_Data (se_key.s),
__ICFEDIT_SE_Key_region_ROM_end__	0x08000400 + 0x3FF		
__ICFEDIT_SE_Startup_region_ROM_start__	0x08000800		.SE_Startup_Code (se_startup.c，里面就一个函数SE_CORE_Startup)
__ICFEDIT_SE_Code_nokey_region_ROM_start	0x08000800 + 0x100	SE_ROM_region	其他readonly,
__ICFEDIT_SE_Code_region_ROM_end__	0x08004FFF		
__ICFEDIT_SE_IF_region_ROM_start__	0x08005000	SE_IF_ROM_region	.SE_IF_Code (se_interface_xxx.c三个文件,主要是se_interface _bootloader.c里面的函数都要先检查caller是否 来自SFU code),
__ICFEDIT_SE_IF_region_ROM_end__	0x080055FF		
__ICFEDIT_SB_region_ROM_start__	0x08005600	SB_ROM_region	其他readonly,
__ICFEDIT_SB_region_ROM_end__	0x0800DFFF		

工程三（UserApp）的输入输出

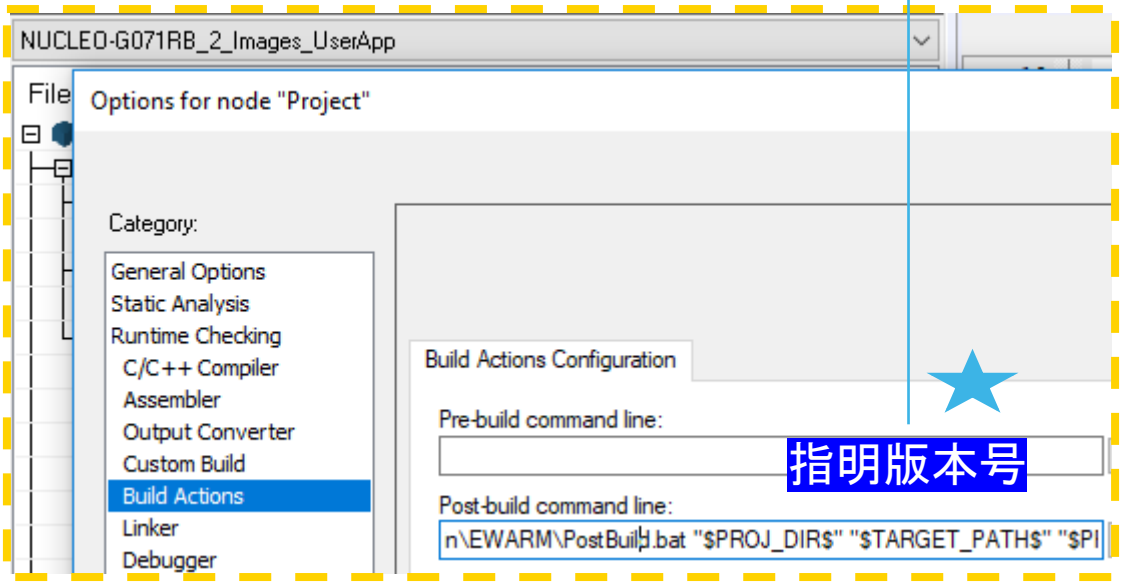


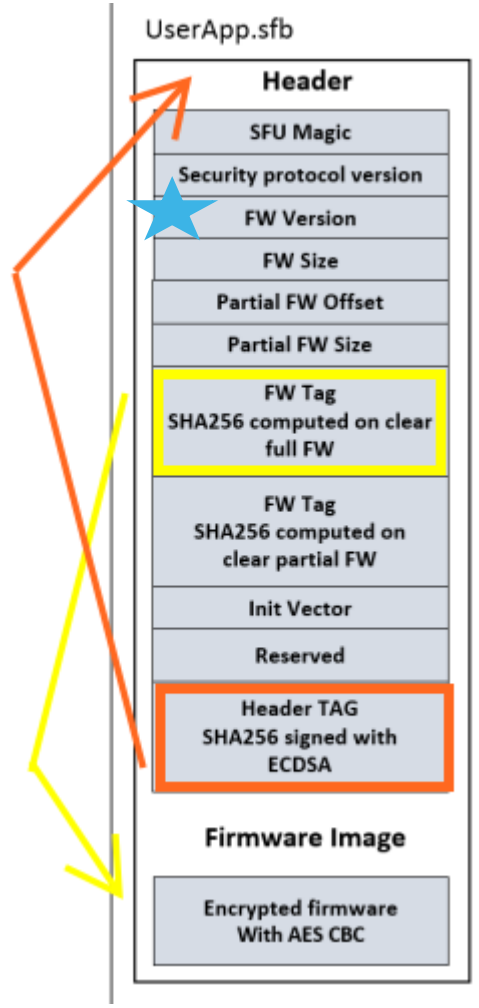
2_Images_UserApp\EWARM\
UserApp.bin

postbuild.bat

2_Images_UserApp\Binary\
UserApph.sfu

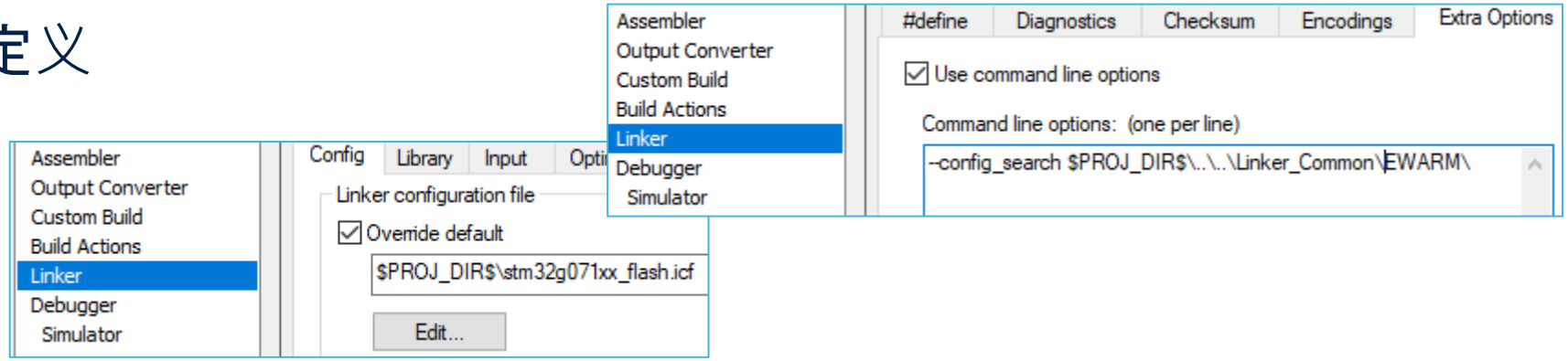
2_Images_UserApp\Binary\
SBSFU_UserApp.bin





工程三（UserApp）的链接文件

- 根据三个linker文件的定义
  - stm32g071_flash.icf
  - mapping_sbsfu.icf
  - mapping_fwimg.icf
- 生成的data和code

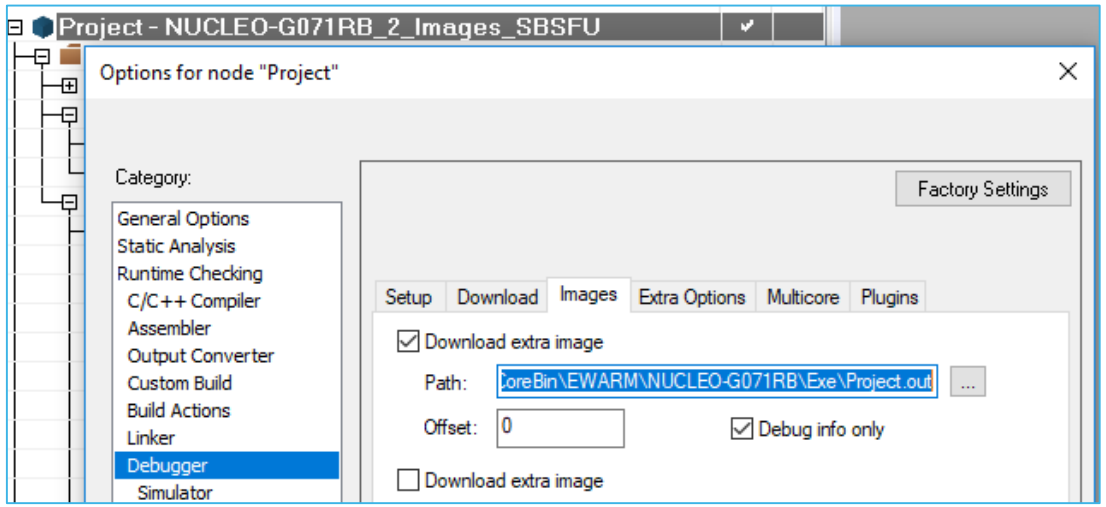


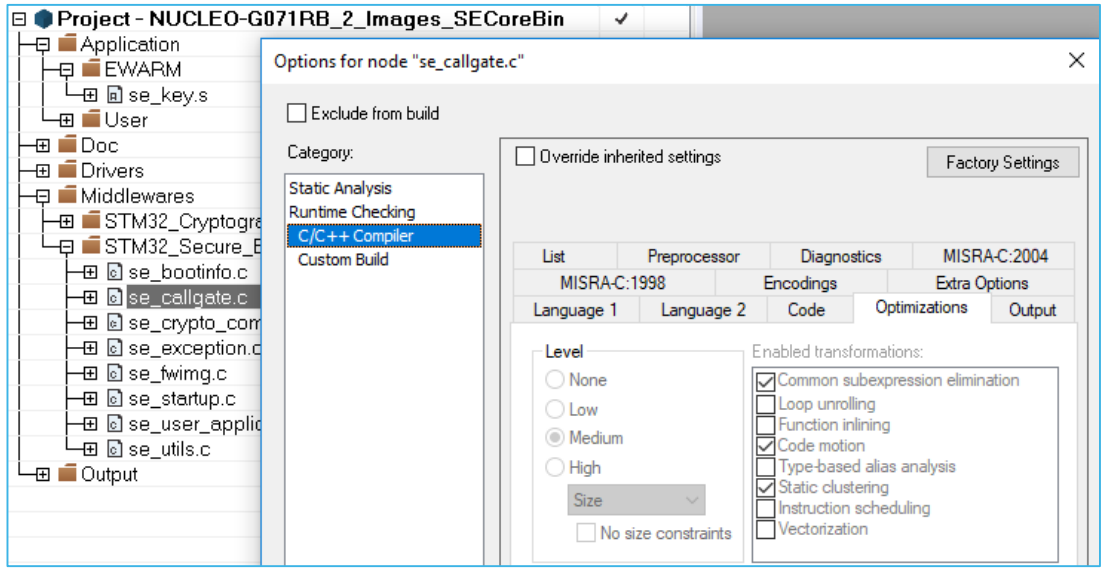
__ICFEDIT_region_SLOT_0_start__	0x0800E000	SLOT_0_region //32K	前0x800/2K是放header，之后才开始放image	
__ICFEDIT_intvec_start__ __ICFEDIT_region_ROM_start__	0x0800E000 + 0x800			ROM_region
__ICFEDIT_region_SLOT_0_end__ __ICFEDIT_region_ROM_end__	0x08015FFF			

__ICFEDIT_SB_region_SRAM1_start__ __ICFEDIT_region_RAM_start__	0x20001000	SB_SRAM1_region	CSTACK, HEAP, CSTACK, HEAP, aes_block_padding // 16字节对齐	RAM_region
__ICFEDIT_SB_region_SRAM1_end__ __ICFEDIT_region_RAM_end__	0x20008FFF			

调试SBSFU：技巧一

- 如何调试SECoreBin部分的代码
- 为了跟踪代码运行，取消全部优化？
- 增加或减少打印信息



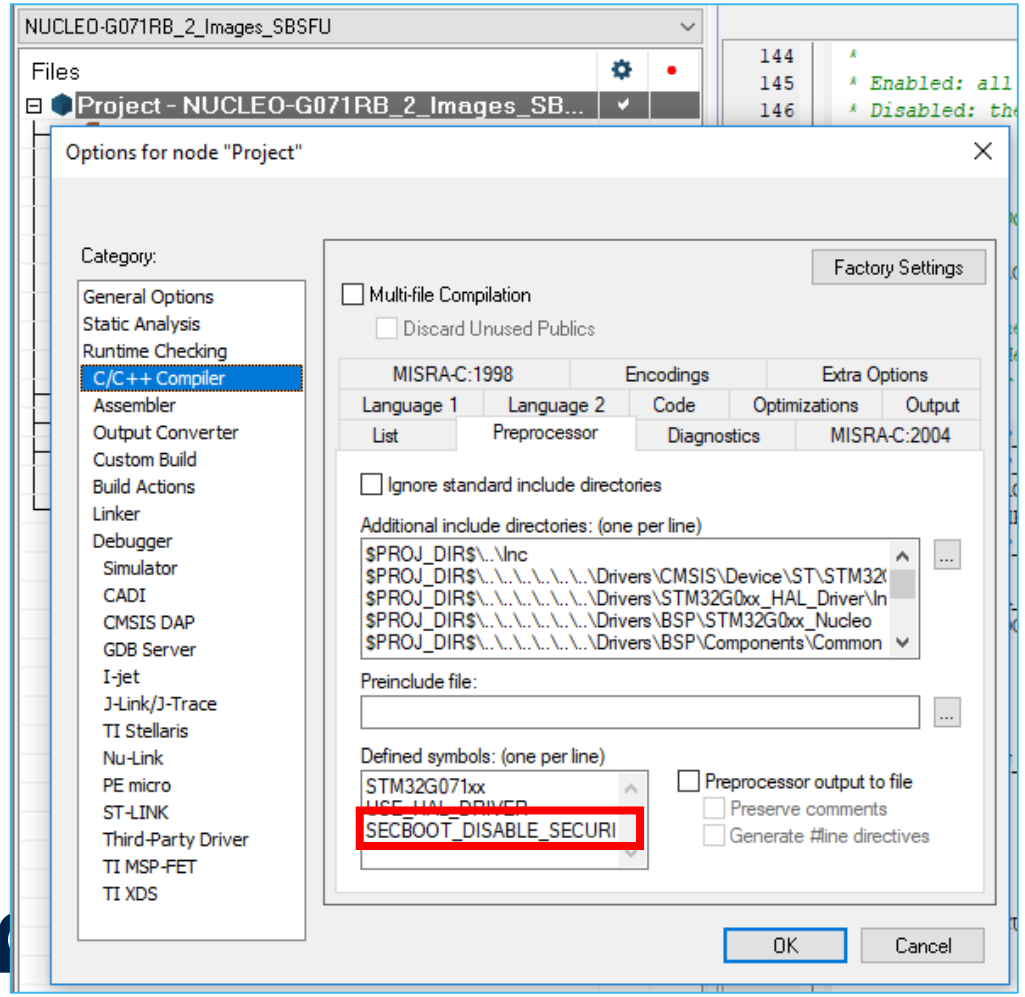


app_sfu.h

```
83  #define SFU_DEBUG_MODE              /*!< Comment this define to optimize memory footprint (debug mode removed)
84                                           No more print on terminal during SBSFU execution */
85
86  /*#define SFU_VERBOSE_DEBUG_MODE*/  /*!< Uncomment this define when in verbose Debug mode.
87                                           this switch activates more debug prints in the console (FSM state info...) */
```

调试SBSFU：技巧二

- 去掉所有保护



app_sfu.h

```
150  /*!< Disable all security IPs at once when activated */
151  /*#define SECBOOT_DISABLE_SECURITY_IPS*/
152
153  #if !defined(SECBOOT_DISABLE_SECURITY_IPS)
154
155  /* Uncomment the following defines when in Release mode.
156     In debug mode it can be better to disable some of the following protection
157     for a better Debug experience (WRP, RDP, IWDG, DAP, etc.) */
158
159  #define SFU_WRP_PROTECT_ENABLE
160  #define SFU_RDP_PROTECT_ENABLE
161  #define SFU_PCROP_PROTECT_ENABLE
162  #define SFU_TAMPER_PROTECT_ENABLE
163  #define SFU_DAP_PROTECT_ENABLE
164  #define SFU_DMA_PROTECT_ENABLE
165  #define SFU_IWDG_PROTECT_ENABLE
166  #define SFU_MPU_PROTECT_ENABLE
167
168  /*!< Only accessible in Secure access mode,the Secure user software is stored in
169   the secure user memory, a configurable protected area which is part of the user
170   main memory. */
171  #define SFU_SECURE_USER_PROTECT_ENABLE
172  /*!< WARNING: Should be enabled at the end of product development and test steps.
173  When enabling this lock, Static protections cannot be modified any more and Debug
174   is finally disabled. */
175  /*#define SFU_FINAL_SECURE_LOCK_ENABLE */
176
177  #if defined(SFU_FINAL_SECURE_LOCK_ENABLE)
178  #define SFU_PROTECT_RDP_LEVEL   (OB_RDP_LEVEL_2)
179  #define SFU_UNIQUE_BOOT_ENTRY
180  #else
181  #define SFU_PROTECT_RDP_LEVEL   (OB_RDP_LEVEL_1)
182  #endif /* SFU_FINAL_SECURE_LOCK_ENABLE */
183
184  #endif /* !SECBOOT_DISABLE_SECURITY_IPS */
```

调试SBSFU：技巧二

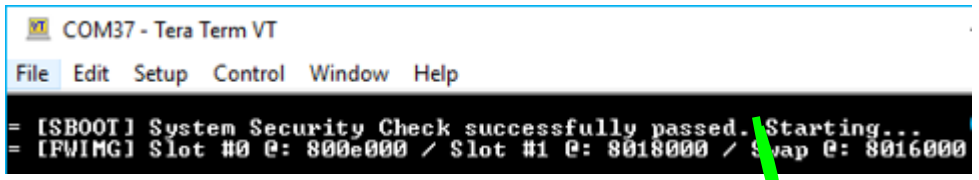
- 去掉所有保护
  - 项目设置Preprocessor
- 根据需要，去掉一些保护
  - SFU_PCROP_PROTECT_ENABLE
  - SFU_DAP_PROTECT_ENABLE
  - SFU_IWDG_PROTECT_ENABLE
  - SFU_RDP_PROTECT_ENABLE
  - 特别注意
    - SFU_FINAL_SECURE_LOCK_ENABLE

app_sfu.h

```
150  /*!< Disable all security IPs at once when activated */
151  /*#define SECBOOT_DISABLE_SECURITY_IPS*/
152
153  #if !defined(SECBOOT_DISABLE_SECURITY_IPS)
154
155  /* Uncomment the following defines when in Release mode.
156     In debug mode it can be better to disable some of the following protection
157     for a better Debug experience (WRP, RDP, IWDG, DAP, etc.) */
158
159  #define SFU_WRP_PROTECT_ENABLE
160  #define SFU_RDP_PROTECT_ENABLE
161  #define SFU_PCROP_PROTECT_ENABLE
162  #define SFU_TAMPER_PROTECT_ENABLE
163  #define SFU_DAP_PROTECT_ENABLE
164  #define SFU_DMA_PROTECT_ENABLE
165  #define SFU_IWDG_PROTECT_ENABLE
166  #define SFU_MPU_PROTECT_ENABLE
167
168  /*!< Only accessible in Secure access mode,the Secure user software is stored in
169   the secure user memory, a configurable protected area which is part of the user
170   main memory. */
171  #define SFU_SECURE_USER_PROTECT_ENABLE
172  /*!< WARNING: Should be enabled at the end of product development and test steps.
173  When enabling this lock, Static protections cannot be modified any more and Debug
174   is finally disabled. */
175  /*#define SFU_FINAL_SECURE_LOCK_ENABLE */
176
177  #if defined(SFU_FINAL_SECURE_LOCK_ENABLE)
178  #define SFU_PROTECT_RDP_LEVEL   (OB_RDP_LEVEL_2)
179  #define SFU_UNIQUE_BOOT_ENTRY
180  #else
181  #define SFU_PROTECT_RDP_LEVEL   (OB_RDP_LEVEL_1)
182  #endif /* SFU_FINAL_SECURE_LOCK_ENABLE */
183
184  #endif /* !SECBOOT_DISABLE_SECURITY_IPS */
```

life.augmented

检查安全环境配置



- **静**态配置
  - 设置选项字节来实现的保护
  - **可通**过宏定义在编译时激活
  - **系**统启动后，SBSFU检查选项字节，根据两**种使用**场景，有不同的处理方式
    - **开**发模式：若没有正确的设置，则SBSFU**会去做相**应设置，然后复位

```
app_sfu.h
214  #define SECBOOT_OB_DEV_MODE
```

    - **生**产模式：若没有正确的设置，SBSFU**复位**，**客**户需要手动做设置，SBSFU**中不会自**动去设置选项字节

- 动态配置
  - **静**态配置检查完成后，检查动态配置
  - **可通**过宏定义在编译时激活
  - **依次**检查
    - MPU
    - **防火**墙
    - WDG
    - Tamper
  - **关掉**GPIO**的**调试复用功能
  - **关掉**DMA时钟

```
app_sfu.h
150  /*#define SECBOOT_DISABLE_SECURITY_IPS*/ /
151
152  #if !defined(SECBOOT_DISABLE_SECURITY_IPS)
153
154  /* Uncomment the following defines when in
155     In debug mode it can be better to disab
156     for a better Debug experience (WRP, RDP
157
158  #define SFU_WRP_PROTECT_ENABLE
159  #define SFU_RDP_PROTECT_ENABLE
160  #define SFU_PCROP_PROTECT_ENABLE
161  #define SFU_TAMPER_PROTECT_ENABLE
162  #define SFU_DAP_PROTECT_ENABLE
```

继续深入探索...

- Secure User memory如何激活
- SE的interface、calling gate，是如何在MPU助力下实现的

谢　谢

life.augmented