M467 Secure ISP with Key Store

NuMicro® 32位系列微控制器範例代碼介紹

文件資訊

應用簡述	本範例程式為 M460_ISP_HID 之進階應用，M467 透過 ISP Programming tool 下載加密韌體後，使用 Key Store 搭配 AES crypto 還原韌體內容並進行更新，藉此完成安全下載。
BSP 版本	M460 Series BSP CMSIS V3.00.002
開發平台	NuMaker-M467HJ V1.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller and microprocessor based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1. 概述

本範例程式提供用戶使用 USB_HID 介面進行韌體更新時，可先透過新唐的 CryptoTool 進行韌體加密，再透過新唐 NuMicro_ISP_Programming_Tool 將加密後的韌體寫入待更新的 NuMaker-M467HJ 開發版。M467 在確認接收完成後會呼叫 XOM 內部的解密 API，搭配硬體加密加速器，以及儲存在 KeyStore Flash 的 AES 密鑰將加密韌體進行解密，而後將韌體寫入 APROM 完成更新流程。

1.1 AES 加密

韌體加密採用 Advanced Encryption Standard (AES) 進行，為一種對稱式加解密算法，用作加解密的金鑰為同一把，長度可為 128、192 或者 256 bits。方式為將要加密的資料分為數個固定大小的區塊進行加密，其優點為安全性高、速度快，為美國國家標準與技術研究院(NIST)所發布。

目前 NIST 針對 AES 算法提供了多種模式，新唐 CryptoTool 採用 AES-CFB-256 ，在進行檔案加解密前須設定 256 bits 的金鑰，以及 128 bits 的 Initialization Vector (IV)，並指定需做加密檔案的位置，以及加密完檔案的存放路徑及檔名，可參考如下:

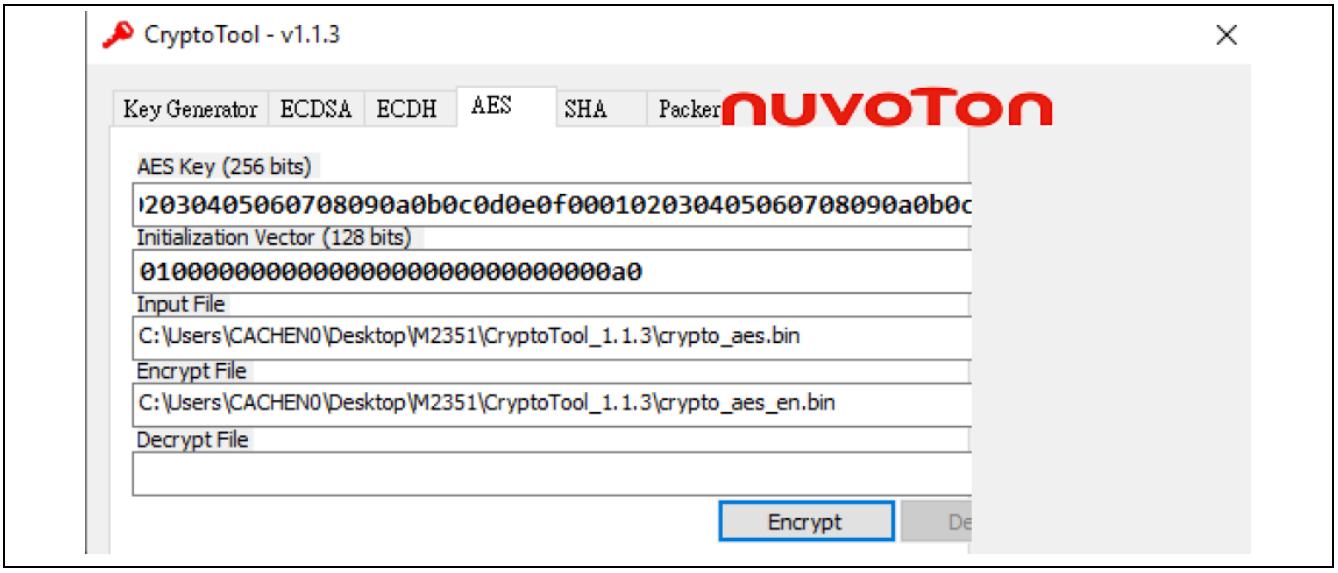


圖 1-1 Nuvoton CryptoTool -v1.1.3

1.2 Execution Only Memory (XOM)

M467 可將內部 APROM 配置成獨立的 XOM 區域，該區域儲存的指令(程式碼)無法被讀取只能執行，最多可配置 4 塊獨立的 XOM，用戶可將機密程式碼放在該區域，防止外部程式意外讀取資料，以及洩漏客戶軟體資訊，詳細說明可參考 M463/M467 Series Technical Reference Manual 章節 6.5.4.6。範例中我們將解密韌體所用到的 API 以及 AES 加解密所使用的 IV 放在該區域內。

1.3 Key Store

M467 Key Store 作為密鑰管理裝置，提供安全存儲的獨立記憶體空間，可將密鑰或重要資料存在 SRAM，Flash 以及 OTP，在進行密鑰存儲設定的過程中，可讓該密鑰只能用在特定硬體加密功能，舉例來說，若設定存在 Flash 的 Key 1 是給 AES 進行使用，則 CPU 以及其他如 ECC 或者 RSA 的硬體加密就無法讀取到該密鑰，讓密鑰的保護達到更為安全及完善。詳細的安全功能及使用說明請參考 M463/M467 Series Technical Reference Manual 章節 6.42。

1.4 軟體架構

1.4.1 Key Store Test

軟體流程共分兩部分來說明，第一部份 Key Store 寫入測試如圖 1-2 所示， 使用 NuMaker-M467HJ ，連接板上 Nu-Link2-me 及測試電腦後，按下 reset 鍵範例程式便開始運行。一開始會先初始化 Key Store，之後便會詢問是否要進行清除 Key Store Flash 的動作，若開發板為第一次使用則不需進行清除，且會看到終端機上顯示” KS FLASH remained size = 4060” 。終端機下一步詢問是否要進行 Key 寫入至 Flash，若第一次使用則必須進行此動作，完成後會將已寫入至 Key Store Flash 的金鑰進行 AES CFB 模式的加解密測試，並將結果顯示在終端機上。

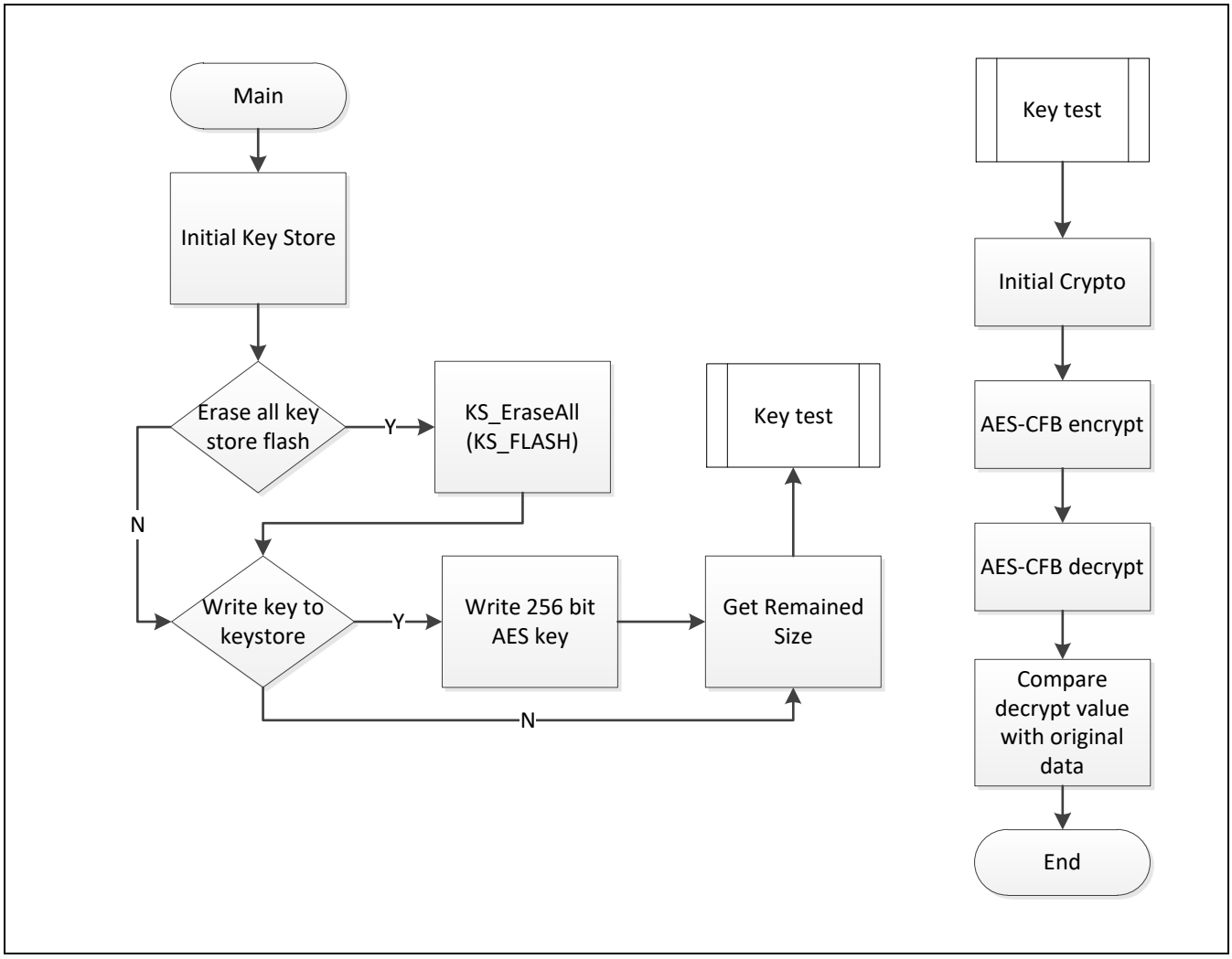


圖1-2 Key test 軟體流程圖

1.4.2 Secure ISP 更新流程

第二部份為主要更新流程，共分兩部分來說明，第一部份先初始化 USB 以及開啟 ISP 功能，之後偵測 PC.13 的 IO 準位來判斷是否進入 ISP 更新程序，若 PC.13 為 Low 則開始下載更新程式，為 high 則跳轉到 APROM 執行。當下載更新完成後便會進到第二部分的 XOM 區域，執行解密以及寫入 APROM 的動作，首先初始化加密引擎，接著設定相關密鑰，初始向量以及 DMA 位址後，加密引擎便會開始還原待更新的程式碼，最後將其寫入 APROM。

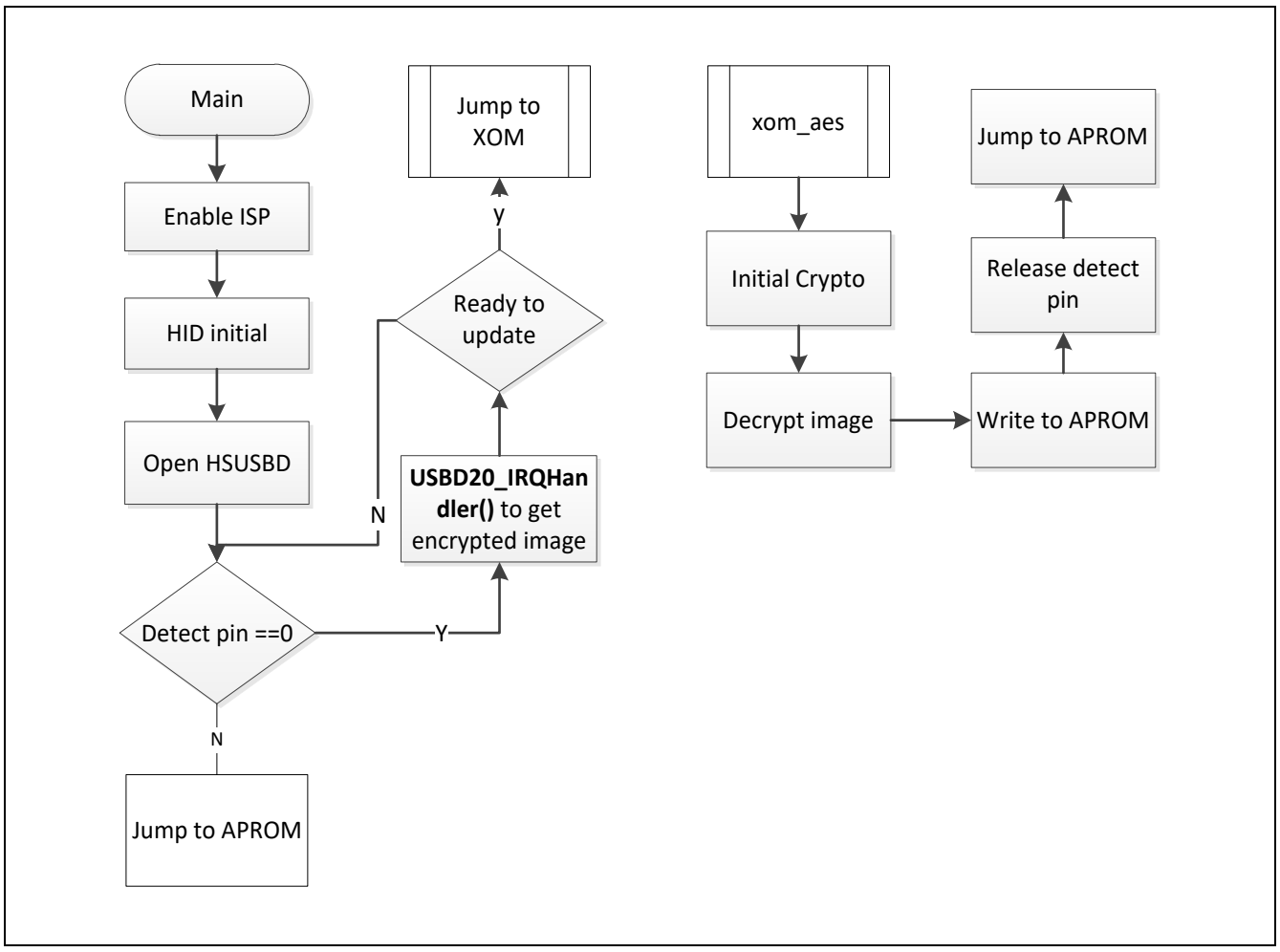

Main
Enable ISP
HID initial
Open HSUSBD
Detect pin ==0
N
Jump to APROM
Y
USBD20_IRQHandler() to get encrypted image
Ready to update
N
y
Jump to XOM
xom_aes
Initial Crypto
Decrypt image
Write to APROM
Release detect pin
Jump to APROM

圖 1-3 軟體流程圖

1.5 功能介紹

1. 將 NuMaker-M467HJ 連接 PC 後，開啟 Keil M467_KS_KeyTest 範例，完成編譯後下載至開發版上，執行 AES key 的寫入測試。

```
KS FLASH remaind size = 4060
Erase all keystore flash?(y/n)
KS FLASH remaind size = 4060
write key to KeyStore?(y/n)
KS FLASH remind size: 4024
The key in key store:
key number is: 1
0x0000  03 02 01 00 07 06 05 04 0b 0a 09 08 0f 0e 0d 0c   ................
0x0010  03 02 01 00 07 06 05 04 0b 0a 09 08 0e 0e 0d 0c   ................

The input data:
0x0000  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff   .."3DUfw........
0x0010  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff   .."3DUfw........

AES encrypt done. The output data:
0x0000  ed 62 a0 fe f1 da e1 41 cc a7 ba 2b 91 9d 19 49   .b.....A...+...I
0x0010  a1 77 bd 71 7b 7f e1 53 65 bf f7 64 11 7e d1 1e   .w.q{..Se..d.~..

AES decrypt done. Decode Data:
0x0000  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff   .."3DUfw........
0x0010  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff   .."3DUfw........

Done!
```

圖 1-4 AES key 測試流程

2. 透過 ISP Programming Tool 下載加密過後的 image，XOM 內的程式碼會將 image 解密後寫入至 APROM 並跳轉到 APROM 執行。

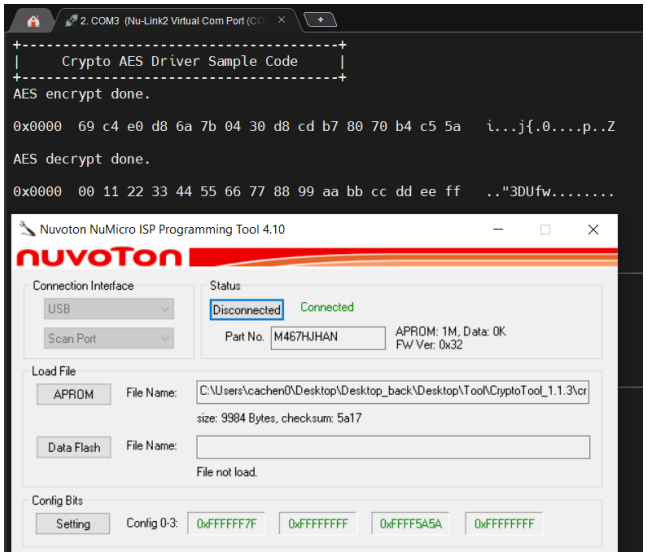


圖 1-5 ISP Programming Tool 更新結果

2. 程式介紹

2.1 Key Store Test

以下程式實現了章節 1.4.1 的架構，核心任務為實現密鑰的寫入及功能測試，其中關鍵的 AES key 透過 au32Key 陣列來設定，用戶在孰悉操作後可自由替換成實際產品所使用的密鑰，並妥善保存。AES 密鑰屬性，建議設定為 CPU 不可讀取，這樣可確保該密鑰只能被 AES engine 所取用，藉此達到完整的保護。AES 密鑰在成功寫入後會返回一編號(i32KeyIdx)，用戶在呼叫 AES engine 搭配 Key Store 做加解密時，需填入正確的編號才能成功還原原始數據。

```
int main(void)
{
    int8_t i8Test = 0;
    int32_t i32KeyIdx = 1;
    /* au32Key value is written into Key Store Flash which is used to decrypt image.
    au32Key value is determined by the user and should be the same with CryptoTool. */
uint32_t au32Key[8] = {
        0x00010203, 0x04050607, 0x08090a0b, 0x0c0d0e0f,
        0x00010203, 0x04050607, 0x08090a0b, 0x0c0d0e0f
    };
    char     chr;

    /* Unlock protected registers */
    SYS_UnlockReg();
    /* Init System, peripheral clock and multi-function I/O */
    SYS_Init();
    /* Init UART for printf */
    UART_Init();
    /* Init Key Store */
    KS_Open();

    printf("KS FLASH remaind size = %d\n", KS_GetRemainSize(KS_FLASH));

    printf("Erase all keystore flash?(y/n)\n");
    chr = getchar();
    if((chr == 'y') || (chr == 'Y')) {
        KS_EraseAll(KS_FLASH);
        printf("KS FLASH remaind size = %d\n", KS_GetRemainSize(KS_FLASH));
    }
    printf("write key to KeyStore?(y/n)\n");
    chr = getchar();
    /*---------------------------------------
    *  Key attribute: flash key
    *  Key meta data: AES-256 bits
    *  au32Key: AES Key
    *  Return Value:
     *  i32KeyIdx, the index value should be set
     *  when using crypto encrypt/decrypt with
     *  key store flash key
    *---------------------------------------*/
    if((chr == 'y') || (chr == 'Y')) {
        i32KeyIdx = KS_Write(KS_FLASH, KS_META_256| KS_META_AES, au32Key);
    }
    if(i32KeyIdx < 0) {
        printf("Fail to write key to Key Store!\n");
        printf("KS FLASH remained size = %d\n", KS_GetRemainSize(KS_FLASH));
```

```
        return -1;
    }

    printf("KS FLASH remind size: %d\n", KS_GetRemainSize(KS_FLASH));

    printf("The key in key store:\n");

    printf("key number is: %d\n", i32KeyIdx);
    DumpBuff((uint8_t *)&au32Key[0], 32);
    /* use AES_Test to check the encryption and decryption with Key Store */
    i8Test = AES_Test(CRPT, KS_FLASH, i32KeyIdx);
    if(i8Test == 0) {
        printf("Done!\n");
    } else {
        printf("fail!\n");
    }
    while(1) {}
}
```

2.2 ISP_HID_20_SecureUpdate

以下程式實現了章節 1.4.2 的架構，前半段透過 USB 以及 ISP 協議，將加密的韌體寫入到事先規劃好的 SRAM 區域(參考 *fw_image.c* 以及 *scatter.sct*)，後半部在 void UpdateRoutine(void) 實現了安全更新的細節，並透過 XOM 進行保護，讓駭客無法得知詳細流程指令。Secure ISP 程式提供給客戶一個安全更新的基本範例，客戶可根據其具體需求，改寫 UpdateRoutine 的流程，如更動使用的 KEYNUM，或者調整 AES 加解密模式以及 IV 值，藉此來完善屬於產品本身的更新機制。

```
int32_t main(void)
{
    /* Unlock write-protected registers */
    SYS_UnlockReg();

    /* Init System, peripheral clock and multi-function I/O */
    if(SYS_Init() < 0) {
        goto _APROM;
    }

    /* Enable ISP */
    CLK->AHBCLK0 |= CLK_AHBCLK0_ISPCKEN_Msk;

    FMC->ISPCTL |= FMC_ISPCTL_ISPEN_Msk | FMC_ISPCTL_APUEN_Msk | FMC_ISPCTL_ISPFF_Msk;

    /* Get APROM and Data Flash size */
    g_u32ApromSize = GetApromSize();
    GetDataFlashInfo(&g_u32DataFlashAddr, &g_u32DataFlashSize);
    if(DetectPin != 0) {
        goto _APROM;
    }

    /* Open HSUSBD controller */
    HSUSBD_Open(NULL, NULL, NULL);

    /* Endpoint configuration */
    HID_Init();

    /* Start transaction */
    HSUSBD->OPER = HSUSBD_OPER_HISPDEN_Msk;    /* high-speed */
    HSUSBD_CLR_SE0();
```

```
    g_u8ISPDone = FALSE;
    /* check PC.13 input status */
    while(DetectPin == 0) {
        /* Polling HSUSBD interrupt flag */
        USBD20_IRQHandler();

        if(g_u8UsbDataReady == TRUE) {
            ParseCmd((uint8_t *)g_u8UsbRcvBuff, 64);
            EPA_Handler();
            g_u8UsbDataReady = FALSE;
        }
        if(ISPDone == TRUE) {
            UpdateRoutine();
            g_u8ISPDone = FALSE;
        }
    }

_APROM:
    /* Reset system and boot from APROM */
    /*set vector to APROM "with IAP", only use with IAP need to set vector */
    FMC->ISPCMD = FMC_ISPCMD_VECMAP;              /* ISP command */
    FMC->ISPADDR = 0x00000000;                    /* Vector remap address */
    FMC->ISPTRG = FMC_ISPTRG_ISPGO_Msk;           /* Trigger ISP command */
    while(FMC->ISPTRG & FMC_ISPTRG_ISPGO_Msk) ;   /* Wait for ISP command done. */
    /*mcu reset and start from APROM*/
    SYS->IPRST0 |= SYS_IPRST0_CPURST_Msk;
    /* Trap the CPU */
    while(1);
}

/* Secure fw update procedure */
void UpdateRoutine(void)
{

    uint32_t StartAddress = 0x0000,  TotalLen;
    TotalLen = g_au32Encryptedimage[0];

    /*---------------------------------------
    *  Step 1, decrypt image
    *  AES-256 CFB mode decrypt fw image
    *---------------------------------------*/
    SYS_UnlockReg();
    KS_Open();
    AES_Open(CRPT, 0, 0, AES_MODE_CFB, AES_KEY_SIZE_256, AES_IN_OUT_SWAP);
    /* Use key in key store */
    AES_SetKey_KS(CRPT, KS_FLASH, KEYNUM);
    AES_SetInitVect(CRPT, 0, (uint32_t*)au32MyAESIV);
    AES_SetDMATransfer(CRPT, 0, (uint32_t)&g_au32Encryptedimage[1],
(uint32_t)au8OutputData, TotalLen);
    AES_Start(CRPT, 0, CRYPTO_DMA_ONE_SHOT);
    while(((CRPT)->AES_STS & CRPT_AES_STS_BUSY_Msk));

    /* Step 2, update APROM */
    FMC_ENABLE_AP_UPDATE();
    WriteData(StartAddress, StartAddress + TotalLen, (uint32_t *)au8OutputData);

    return;
}
```

3. 軟硬體需求

3.1 軟體需求

- BSP 版本
  - M460 Series BSP CMSIS V3.00.002
- IDE 版本
  - Keil uVersion 5.39

3.2 硬體需求

- 電路元件
  - NuMaker-M467HJ V1.0
- 線路示意圖
  - 杜邦線*1 & 連接腳位(PC.13)

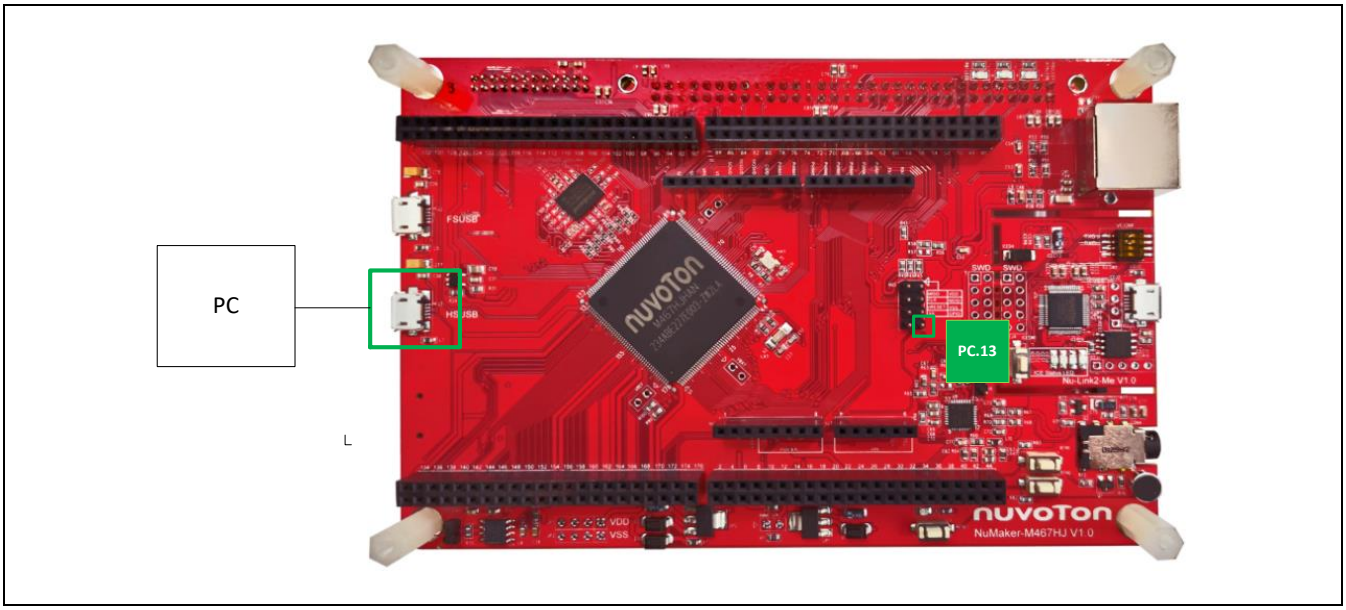


Figure 3-1 线路示意图

4. 目錄資訊

目錄架構如下所示。

EC_M467_Secure_ISP_with_Key_Store_V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) by ARM® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	
Project	Source file of example code

圖 4-1 目錄資訊

5. 範例程式執行

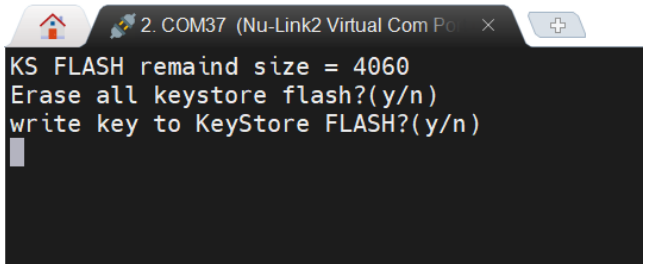
5.1 測試流程

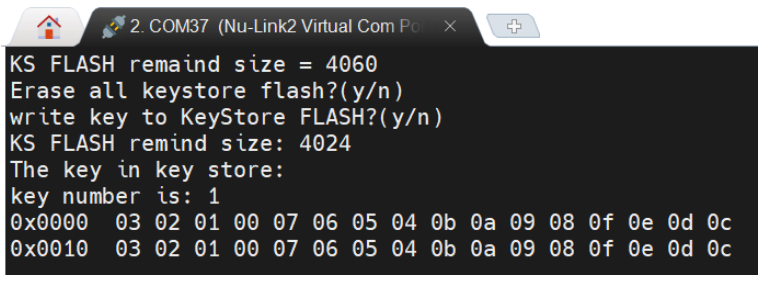
5.1.1 Key 設定及測試

此範例將用於解密更新程式碼的 AES 密鑰寫入 Key Store 的 Flash 區域中，並設定為 AES 硬體加密專用，藉此保護密鑰。

1. 打開 M467_KS_KeyTest 的 KEIL 資料夾，雙擊 *M467_KS_KeyTest.uvprojx*。
2. 編譯範例程式後燒入至 NuMaker- M467HJ 並打開終端機裝置顯示列印訊息。
3. 一開始顯示 Key Store Flash 剩餘空間，若為第一次使用則為 4060 bytes，不須做 Erase all key store flash 的操作。

4. 下一步會詢問是否要進行 Key Store 寫 key 的動作，輸入"y"或"Y"後會將範例程式內 au32Key 的數據寫入，成功寫入後會回傳存放位置的編號，以及打印 au32Key 數據。





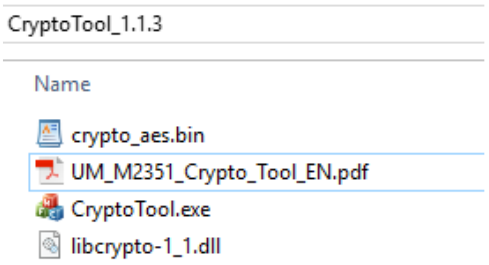
5. Key Store 寫入完成後開始搭配 Crypto engine 進行 AES-CFB 測試，進行一次加密以及解密操作，比較完成後的數據與原始數據是否一致。

```
The input data:
0x0000  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff
0x0010  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff

AES encrypt done. The output data:
0x0000  09 c4 84 cf 9f da 7b e7 6b 00 71 2a c8 e3 78 bd
0x0010  0c 38 3d e8 5b 48 8b e9 18 43 62 e6 00 14 63 1c

AES decrypt done. Decode Data:
0x0000  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff
0x0010  00 11 22 33 44 55 66 77 88 99 aa bb cc dd ee ff

Done!
```

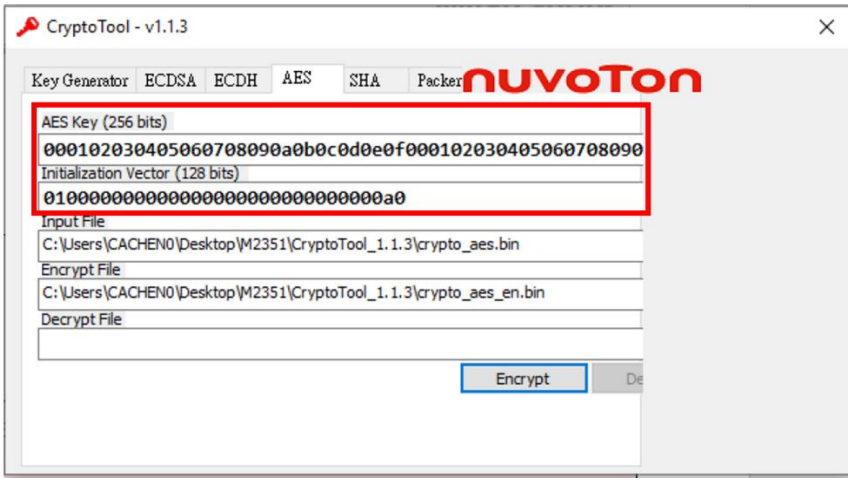
5.1.2 Crypto tool 加密更新檔

新唐 Crypto tool 可至以下連結下載: [Nuvoton NuTool Crypto Tool]

1. 下載後執行 CrptoTool.exe。

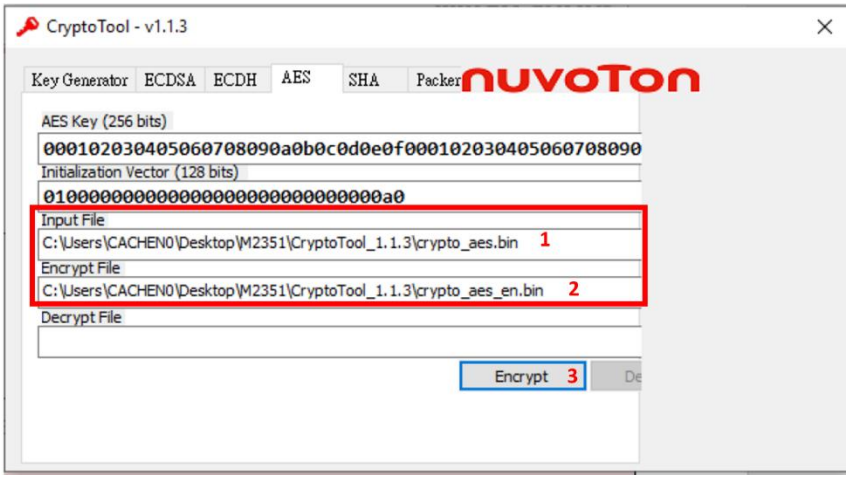


2. 選擇 AES 欄位，設定 AES key 以及 Initialization Vector。



3. 編譯 M460 BSP 內任意一範例程式，並將編譯完成的 bin 檔複製到 CryptoTool 底下。

4. 設定 Input File 及 Encrypt File 的路徑，按下 Encrypt 按鈕，幾秒後會在該路徑下看到生成的加密 bin 檔。



5. 完成

5.1.3 ISP 程式下載及 XOM 設定

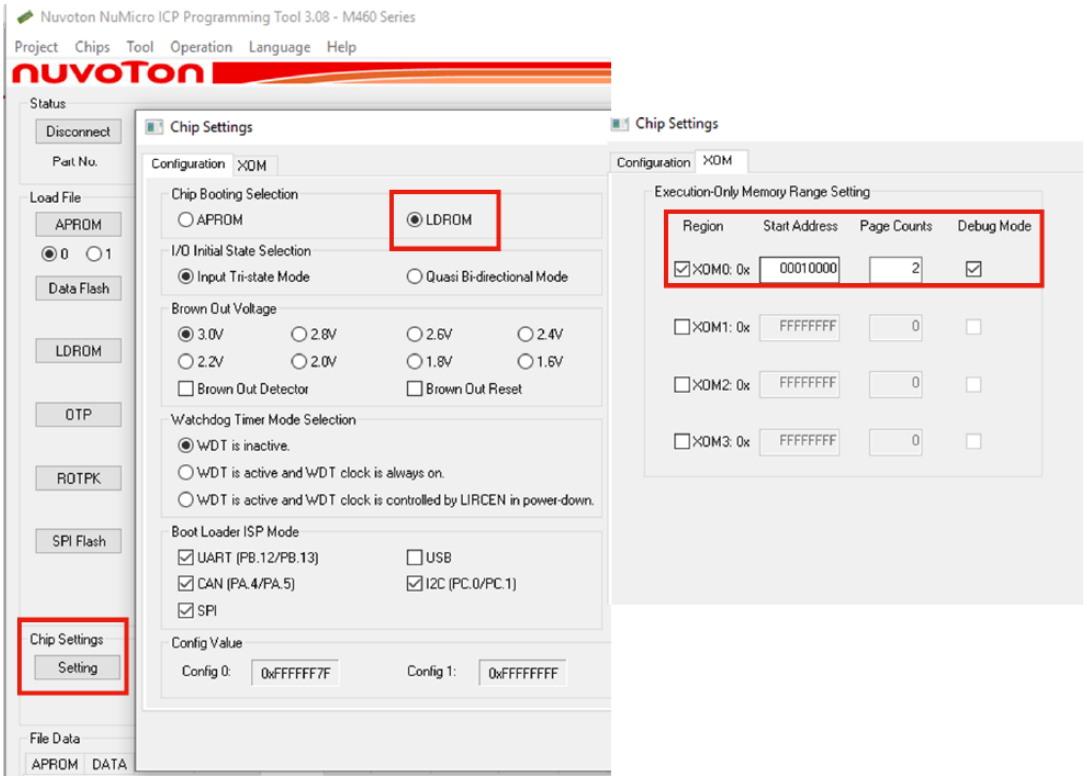
1. 根據目錄資訊章節進入 ExampleCode 路徑中 M467_ISP_HID_20_SecureUpdate 的 KEIL 資料夾，雙擊 *M467_ISP_HID_20_SecureUpdate.uvprojx*。

2. 編譯後按下 Download 寫入至開發版。

3. 打開 Nuvoton NuMicro_ICP_Programming_Tool 點選 M460 series 連接開發版。

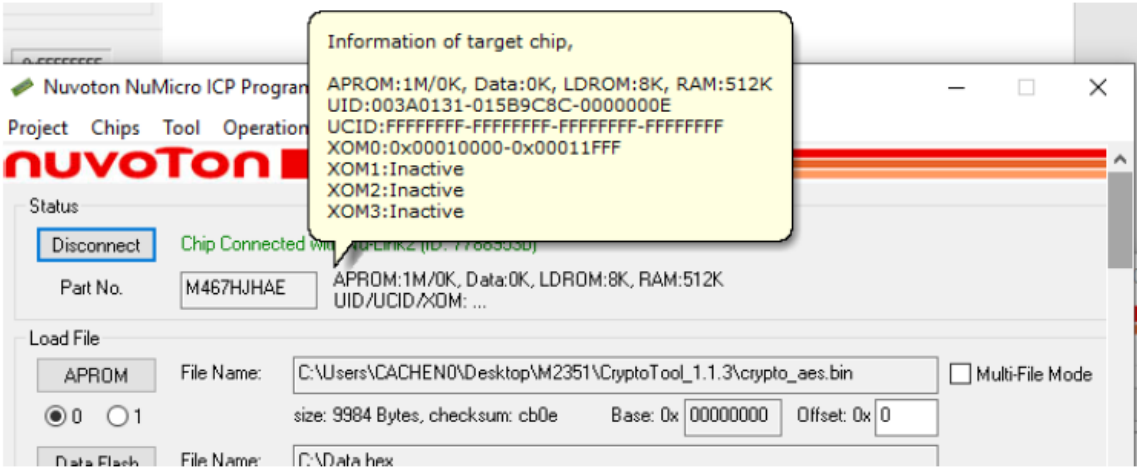
4. 點選 Chip Settings -> Chip Booting Selections 設定 LDROM。

5. 點選 Chip Settings -> XOM 依照下圖紅框設定 XOM，Debug Mode 不勾選則在 debug 模式下無法執行單步除錯且看不到指令內容。



6. 確認 Programing 區域只勾選 Chip Setting 後按下 Start。

7. 重新連上 Tool，確認 XOM0 啟動。



5.1.4 ISP 安全範例更新步驟執行

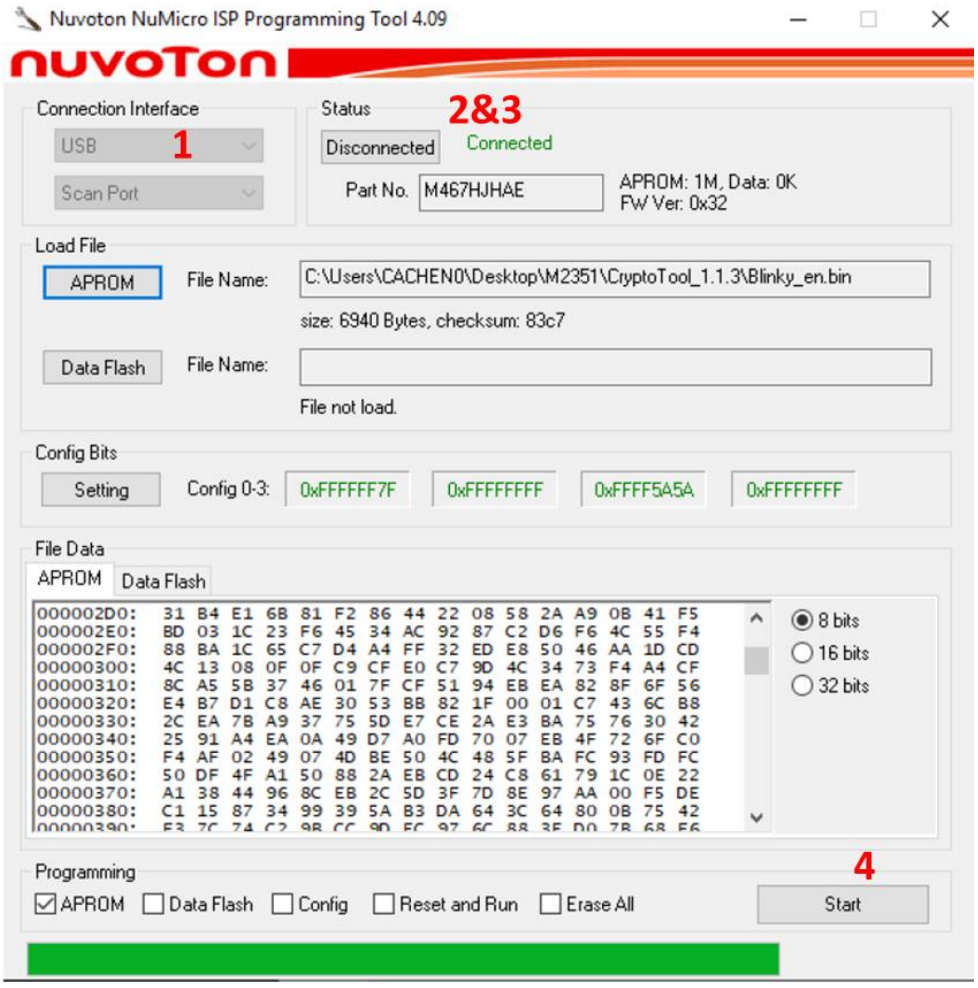
完成上述步驟後，本小節開始說明如何透過新唐 ISP Tool 更新程式碼

1. 開啟 NuMicro_ISP_Programming_Tool，Connection interface 選擇 USB。

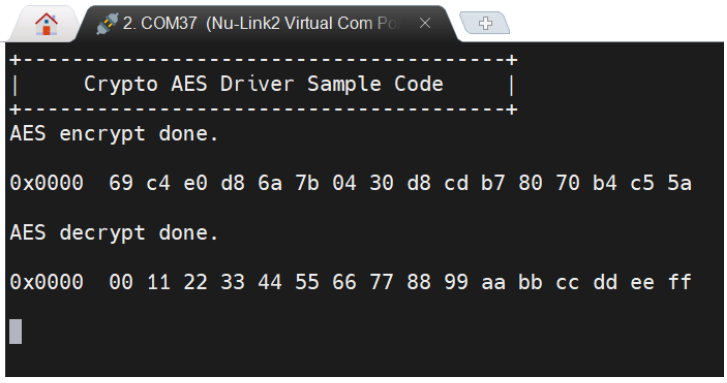
2. 連接開發版 HSUSB port 及 PC 並將 PC.13 短路到地。

3. 按下 Connect 後按下開發板上的 Reset 按鈕，確認連上 tool。

4. 點選 APROM 並載入剛剛加密完成的更新檔，勾選 Programing 區域 APROM 後按下 Start。



5. 將 PC.13 改連接到 V_{CC}，當程式偵測到輸入訊號由低準位拉到高準位後，會自動跳轉到 APROM 執行，可在終端機上看到程式執行畫面。



6. 修訂紀錄

Date	Revision	Description
2024.3.29	1.00	初始發布。

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.