AN0128
Application Note

Getting Started with AT32F435 & AT32F437

Introduction

This application note is used to help users quickly develop projects using AT32F435xx/AT32F437xx, where AT32F437 is designed with EMAC function.

Note: The corresponding code in this application note is developed on the basis of V2.x.x BSP provided by Artery. For other versions of BSP, please pay attention to the differences in usage.

Applicable products:

Part number	AT32F435xx
	AT32F437xx

Contents

List of tables

List of figures

1 Development resources

Resources download link:

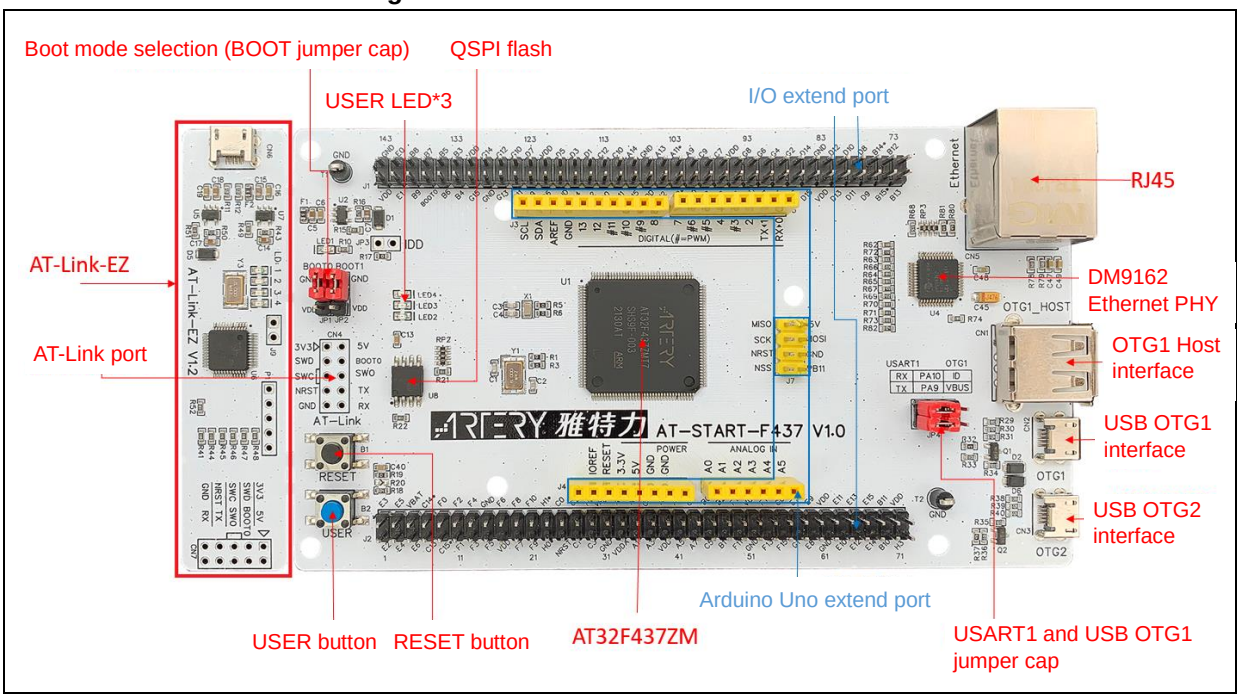
- Visit Artery website: https://www.arterychip.com

1.1 Set up AT32 development environment

1.1.1 Debug tools and evaluation board

The AT32F435 /AT32F437 evaluation board comes with AT-Link-EZ for the debug purpose, as shown in the left red box of the figure below. It can also be disassembled from the evaluation board and used separately with other circuit boards. This debug tool can be used for several purposes such IDE online debugging, online programming and USB-to-serial interface.

Figure 1. AT-START-F437 and AT-Link-EZ



Note: For details about resources for AT-START evaluation board, please refer to UM_AT_START_F43x_Vx.x. Path: ARTERY's official website→PRODUCTS→High performance→AT32F4xx; download and unzip Evaluation Board package, and get the "\AT_START_F43x_Vx.x\03_Documents".

Figure 2. AT-START-F437 evaluation board package

Evaluation Board

Download	Description	Version
AT-START-F435	AT32F435 evaluation board supporting Arduino standard interfaces	V1.10

1.1.2 Programming tools and software resources

- AT programming tools and software:AT-Link / AT-Link+ /AT-Link-Pro / AT-Link-ISO /AT-Link-EZ, ICP/ISP

- 3rd party programming tools: J-Link, Armfly, Alientek, XWOPEN, ICWORKSHOP, ZLG, MaxWiz, Amomcu, Acroview, Forcreat, Galecomm, Prosystems, Rx-prog, Sinaen, XELTEK, Zhifeng, etc.

Note: For more information, please visit ARTERY's official website→SUPPORT→Hardware Development Tool and 3RD Party Writer.

- For ICP usage instructions, please refer to *UM_ICP_Programmer*. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4xx; download and unzip ICP tool, and get the "Artery_ICP_Programmer_Vx.x.xx\Document\UM_ICP_Programmer".
- For ISP usage instructions, please refer to *UM_ISP_Programmer*. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4xx; download and unzip ISP tool, and get the "Artery_ISP_Programmer_Vx.x.xx\Document\UM_ISP_Programmer".
- For AT-Link usage instructions, please refer to *UM0004_AT-Link_User_Manual*. Path: ARTERY's official website→PRODUCTS→High performance→AT32F4xx; download and unzip AT-Link-Family, and then get the "AT_Link_CH_Vx.x.x\05_Documents\UM0004_AT-Link_User_Manual_EN_Vx.x.x".

Figure 3. ICP/ISP/AT-Link-Family

Tool

Download	Description	Version
AT32 IDE_Linux AT32 IDE_Windows	A software development environment for cross-platform ARM embedded system based on Eclipse development supporting AT32 MCU	V1.0.05
AT-Link	Emulation and online/offline programming tools supporting AT32 MCU	V2.1.1
AT-Link Console_Linux AT-Link Console_Windows	In-Circuit-Programming Console tool supporting AT32 MCU	V3.0.07
ICP	In-Circuit-Programming tool supporting AT32 MCU	V3.0.10
ISP	In-System-Programming tool supporting AT32 MCU	V2.0.10
ISP_Multi-Port	In-System-Multi-Port Programming tool supporting AT32 MCU	V2.0.10
ISP Console_Linux ISP Console_Windows	In-Circuit-Programming Console tool supporting AT32 MCU	V3.0.07

1.1.3 AT32 development environment

1.1.3.1 Template projects

The commonly used IDE template projects are available from BSP provided by ArteryTek. BSP path: ARTERY's official website→PRODUCTS→High performance→AT32F4xx.

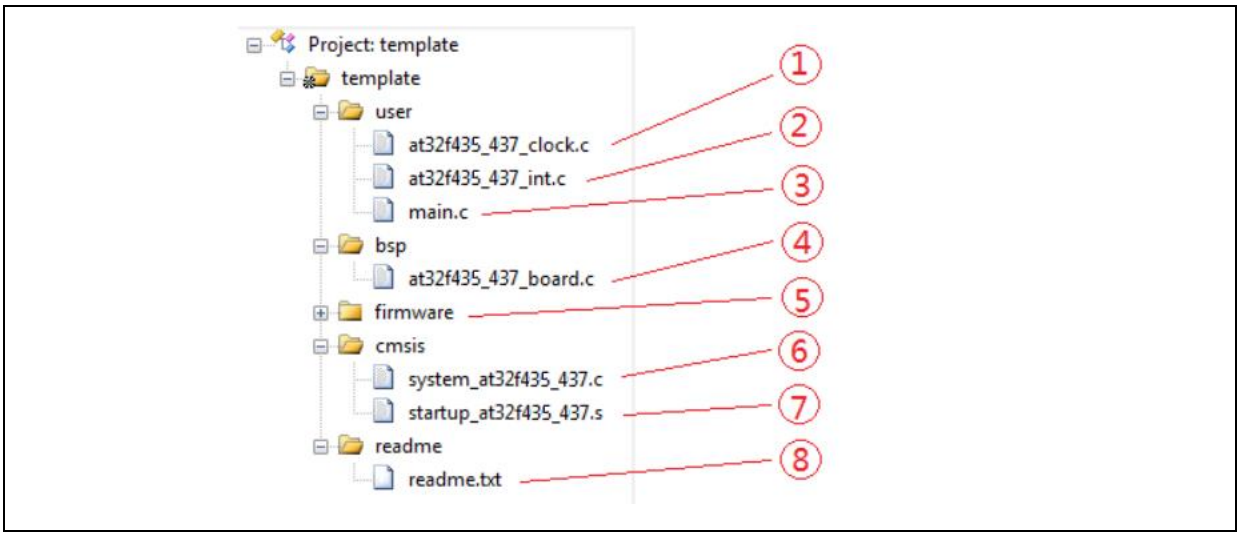
Figure 4. BSP package

BSP

Download	Description	Version
Firmware Library	AT32F435 firmware library BSP user guide	V2.1.2

For the AT32F435/437 series, template projects based on Keil_v5/Keil_v4/IAR_6.10/IAR_7.4/IAR_8.2/eclipse_gcc/at32_ide are contained in the BSP. Path: AT32F435_437_Firmware_Library_V2.x.x\project\at_start_f4xx\templates. Open the project folder and click on the project file to open the corresponding IDE project. The example of Keil_v5 template project is shown below.

Figure 5. Keil_v5 templates



Contents in the project:

① at32f435_437_clock.c: clock configuration file, contains default clock frequency and clock path;

② at32f435_437_int.c: interrupt file, part of core interrupt function code flow is written by default;

③ main.c: main code file of the template project;

④ at32f435_437_board.c: board-level configuration file, contains settings of AT-START on-board buttons and LEDs;

⑤ at32f435_437_xx.c under *firmware* folder: driver file of on-chip peripherals;

⑥ system_at32f435_437.c: system initialization file;

⑦ startup_at32f435_437.s: startup file;

⑧ readme.txt: project documentation, contains application functions, setting methods and associated application notes (ApNote) of the template project.

Except for templates, BSP also includes code examples (Keil_v5 project files) in terms of peripherals for reference.

Path: AT32F435_437_Firmware_Library_V2.x.x\project\at_start_f4xx\examples.

Note: For more details about BSP, please refer to "Section 4 BSP application" of AT32F435_437 Firmware BSP&Pack User Guide. Path: ARTERY's official website→PRODUCTS→High performance→AT32F4xx; download and unzip, and get the "\AT32F435_437_Firmware_Library_Vx.x.x\document".

1.1.3.2 Pack installation

Install Pack and add the AT32 MCU part number to Keil/IAR. You can download Pack from ARTERY's official website→PRODUCTS→High performance→AT32F4xx.

Figure 6. Pack download

Pack

Download	Description	Version
Keil 4 Keil 5	Supports AT32 MCU to run in Keil MDK	V2.2.0 V2.2.3
IAR	Supports AT32 MCU to run in IAR EWARM	V2.1.6
Segger	Supports Segger tools to identify AT32 MCU	V2.0.7

For Keil compiling system, keil 4.74 /5.23 or above is recommended. If Keil_v5 is used, please unzip “Keil5_AT32MCU_AddOn” and install the corresponding ArteryTek.AT32F435_437_DFP. If Keil_v4 is used, please install Keil4_AT32MCU_AddOn By default, the Keil installation path can be recognized automatically during installation. If the path is not recognized or incorrect, you need to manually select the Keil installation path.

Figure 7. Install ArteryTek.AT32F435_437_DFP

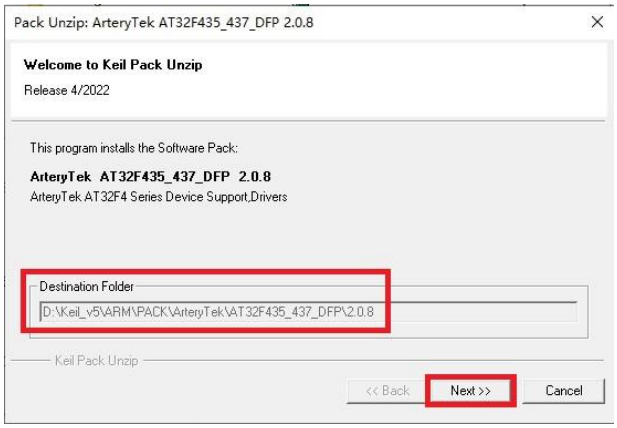
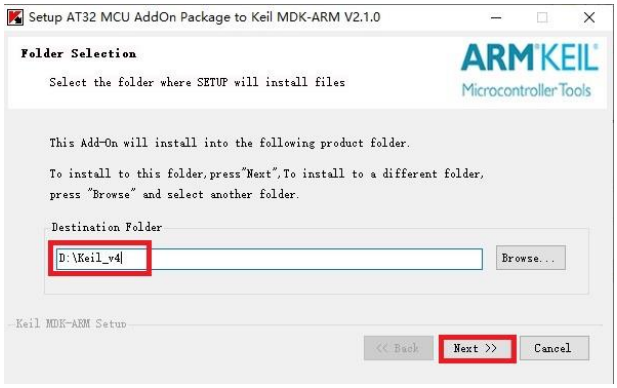
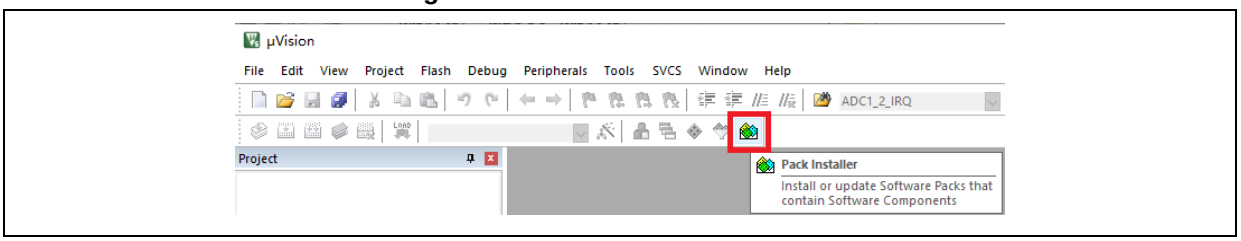


Figure 8. Install Keil4_AT32MCU_AddOn



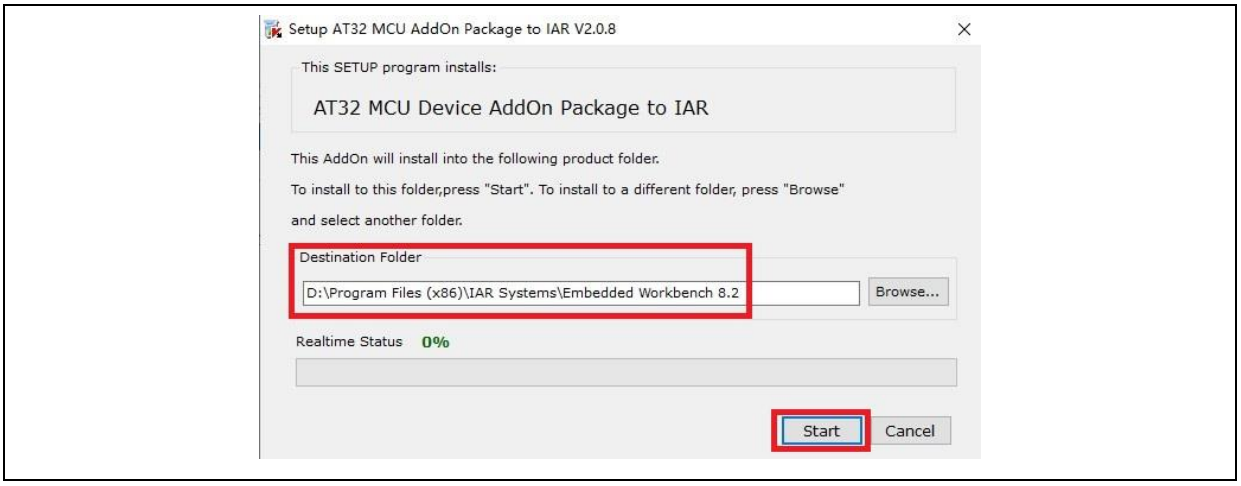
You can also open keil and click on “Pack Installer” icon; then click on the top left “file” and select “import” to import the corresponding pack downloaded from ARTERY's official website.

Figure 9. Pack Installer icon in Keil



For IAR compiling system, IAR7.0 or IAR6.1 above is recommended. It is necessary to install IAR_AT32MCU_AddOn. By default, the IAR installation path can be recognized automatically during installation. If the path is not recognized or incorrect, you need to manually select the IAR installation path.

Figure 10. Install IAR_AT32MCU_AddOn

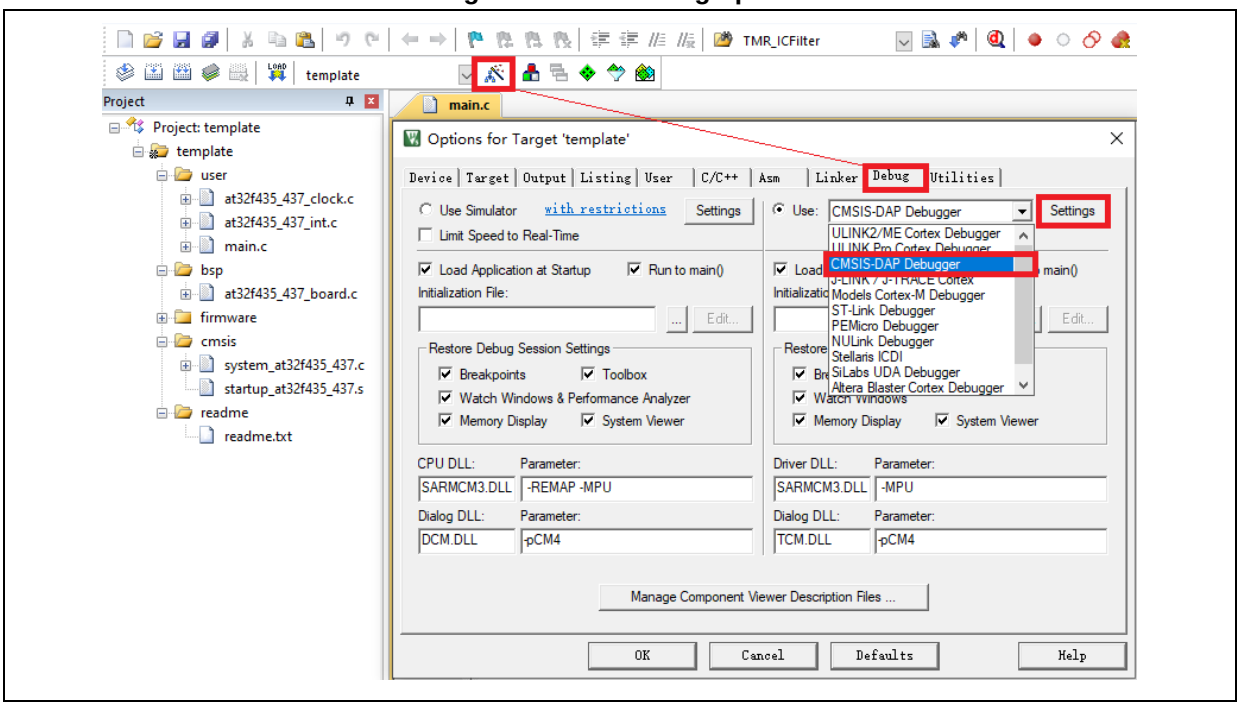


Note: For more details about Pack setup, please refer to "Section 2 Pack setup" of AT32F435_437 Firmware BSP&Pack User Guide. Path: ARTERY's official website→PRODUCTS→High performance→AT32F4xx; download and unzip BSP, and get the "\ AT32F435_437_Firmware_Library_Vx.x.x\document".

1.1.3.3 Use AT-Link for debug and download

If you want to use AT-Link in IAR, select CMSIS-DAP in Debugger option.

Figure 11. Keil Debug option



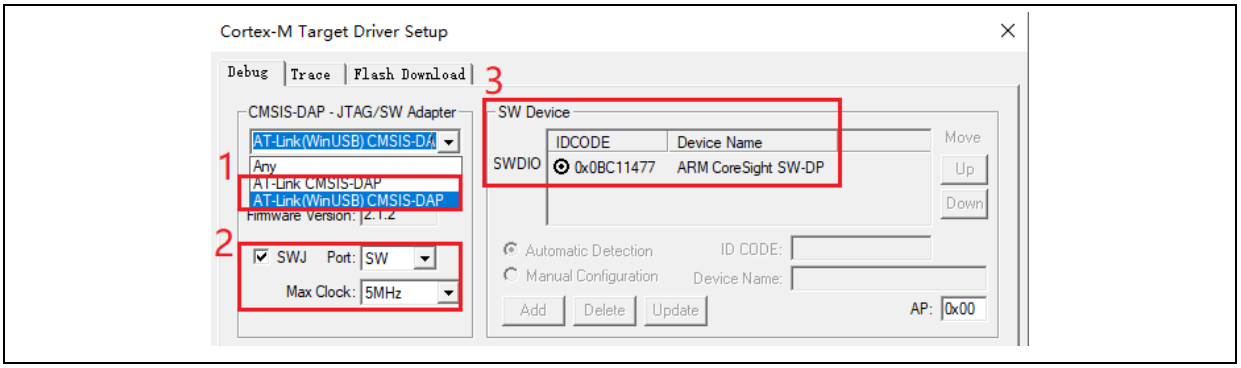
Go to Debug and click on Settings to enter the Cortex-M Target Driver Setup interface.

1. Select AT-Link(WinUSB)-CMSIS-DAP/AT-Link-CMSIS-DAP;

Note: For details about WinUSB, please refer to FAQ0136_How to use AT-LINK WinUSB to improve download speed (ARTERY's official website→SUPPORT→FAQ→FAQ0136).

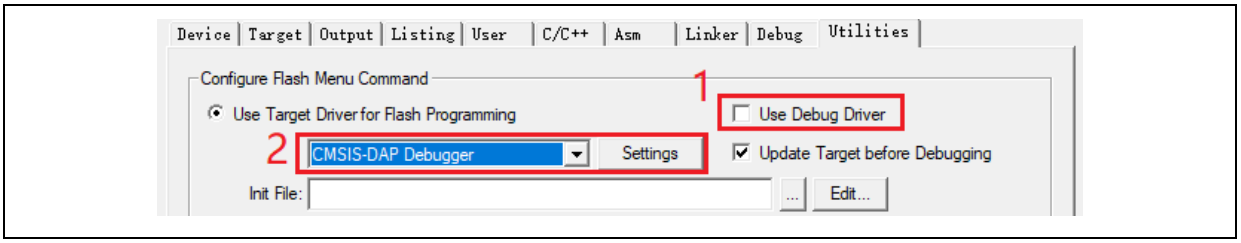
2. Find Port, select SW and then tick SWJ;
3. Confirm that the ARM SW-DP debug module is recognized.

Figure 12. Keil Debug Settings



Click on Utilities and untick option box 1; then select CMSIS-DAP Debugger in option box 2, and finally tick option box 1 (it should be unticked first and then ticked).

Figure 13. Keil Utilities



If you want to use AT-Link in IAR, please click on Project and select Options; then select CMSIS-DAP in Debugger option, and select SWD in CMSIS DAP option.

Figure 14. IAR Debug option

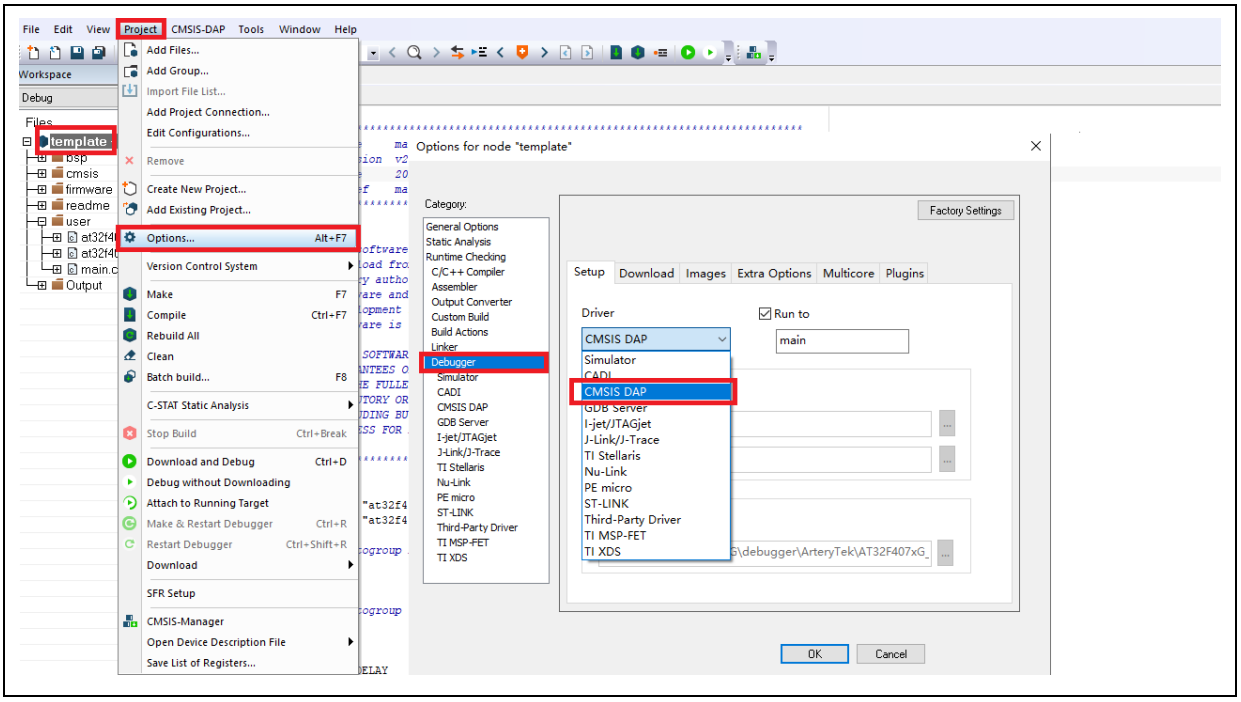
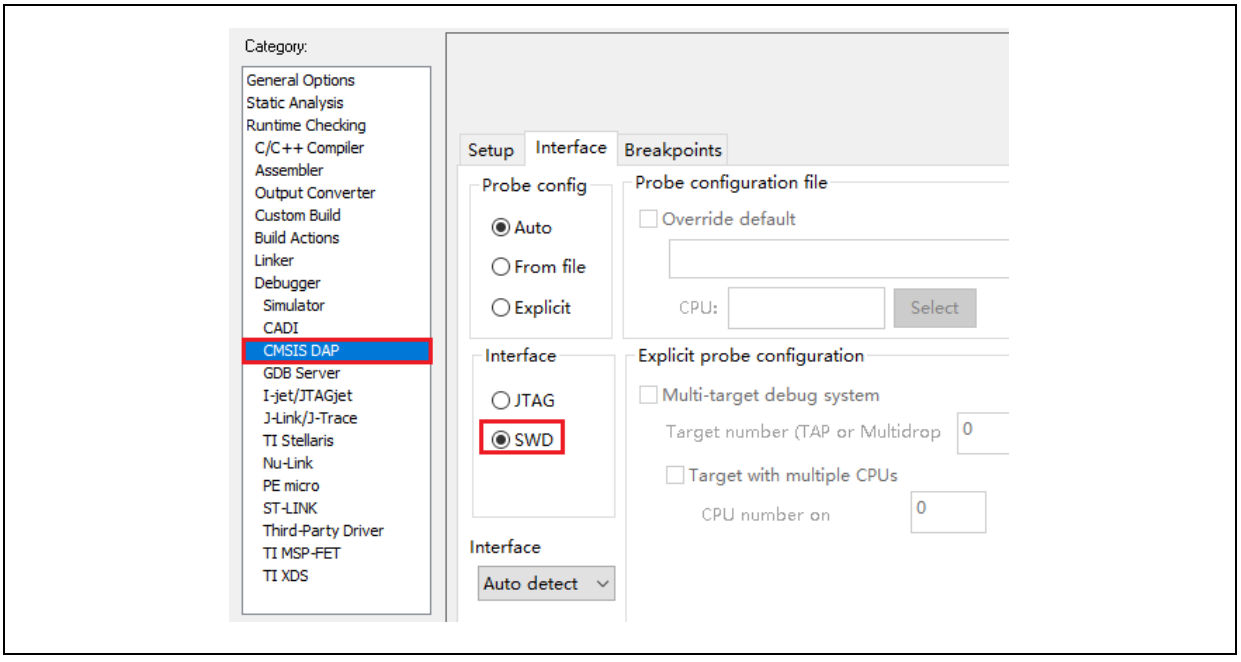


Figure 15. IAR CMSIS-DAP option



Note: For details about Flash algorithm file, MCU switch and solutions for "J-Link cannot find MCU", please refer to AT32F435_437Firmware BSP&Pack User Guide. Path: ARTERY's official website→PRODUCTS→High performance→AT32F4xx; download and unzip BSP, and get the "\AT32F435_437_Firmware_Library_Vx.x.x\document".

1.1.4 How to replace AT32F403A/407

- Please refer to *MG0018_ Migrating from AT32F403A_407 to AT32F435_437*. Path: ARTERY's

official website→PRODUCTS→High performance→AT32F4xx.

- If the program still cannot run properly after completing the above steps, please refer to other sections of this application note or contact the agent and ARTERY technicians for help.

1.2 AT32F435/AT32F437 chip enhanced functions

1.2.1 PLL clock settings

1.2.1.1 PLL setting method

The AT32F435_437 series embed a PLL with a maximum of 288 MHz clock output. The PLL clock configuration register (CRM_PLLCFG) can be used to configure different PLL clock frequencies, with the following formula:

$$\text{PLL output clock} = \text{PLL reference input clock} \times \frac{\text{PLL frequency multiplication factor (PLL\_NS)}}{\text{PLL predivider factor } (\text{PLL}_{\text{MS}}) \times \text{PLL post} - \text{divider factor (PLL\_FR)}}$$

For example, when PLL=288 MHz (HEXT=8 MHz):

```
/*!< config pll clock */
crm_pll_config(CRM_PLL_SOURCE_HEXT, 72, 1, CRM_PLL_FR_2);
```

Where, “HEXT” in the parameter “CRM_PLL_SOURCE_HEXT” indicates that HEXT clock is used as an external clock; PLL_NS=72, PLL_MS=1, and PLL_FR value is “CRM_PLL_FR_2 (0x01, divided by 2).

For details about clock configuration, please refer to *AN0084_AT32F435_437_CRM_Start_Guide*. This document can be downloaded from ARTERY’s official website→SUPPORT→AP Note→AN0084. It introduces how to configure and modify AT32F435/437 clock source, and how to use the New Clock Configuration tool (ARTERY’s official website→PRODUCTS→High performance→AT32F4xx) to quickly generate the desired clock code and apply it to projects.

1.2.1.2 PLL automatic frequency switch feature

When the PLL multiplication frequency of AT32F435 /AT32F437 series is greater than 108 MHz, it is recommended to enable the PLL auto step-by-step frequency switch feature.

Figure 16 gives an example of PLL auto frequency switching function in AT32F435 /AT32F437 BSP.

Figure 16. PLL auto step-by-step switch configurations

```
/* enable auto step mode */
crm_auto_step_mode_enable(TRUE);
/* select pll as system clock source */
crm_sysclk_switch(CRM_SCLK_PLL);
/* wait till pll is used as system clock source */
while(crm_sysclk_switch_status_get() != CRM_SCLK_PLL)
```

```
{
}
/* disable auto step mode */
crm_auto_step_mode_enable(FALSE);
/* update system_core_clock global variable */
system_core_clock_update();
```

Note: If this auto frequency switching function is enabled, it should be disabled right after the completion of clock switching operation. In other words, enabling and disabling must be operated in pair.

1.2.1.3 Flash clock division

The correlation between system clock frequency and Flash clock division is shown below.

System Clock Frequency	Flash Clock Division
<= 240 MHz	FLASH_CLOCK_DIV_2
>240 MHz	FLASH_CLOCK_DIV_3

Code:

```
flash_clock_divider_set(FLASH_CLOCK_DIV_3);          /* Flash divider: system clock divided by 3 */
```

1.2.2 How to use FPU (floating point unit) feature

Please refer to *AN0037_How_to_use_FPU*. This document can be downloaded from ARTERY's official website→SUPPORT→AP Note→AN0037. It introduces how to use AT32 MCU FPU feature and how to configure FPU in Keil / IAR.

1.2.3 AT32F435 /AT32F437 zero-wait/non-zero wait Flash and embedded SRAM configurations

Users can configure the user system data area to allocate the internal Flash and on-chip SRAM.

The on-chip SRAM of AT32F435/ AT32F437 is 384 KB, by default. Users can configure the SRAM to be 128 KB (minimum) or 512 KB (maximum) by setting the EOPB0 bit. **A power-down or system RESET must performed before enabling the EOPB0.**

The user system data area of AT32F435/ AT32F437 is shown below.

Address	Bit	Description
0x1FFF_C010	[7:0]	EOPB0[7:0]: Extended system option Refer the table below for more details. Note: *This bit can be changed only when the security library is disabled.*
	[15:8]	nEOPB0[7:0]: Inverse code of EOPB0[7:0]
	[31:16]	Reserved

EOPB0[7:0]: Extended system option		
256 K Flash	Bit 1:0	00: On-chip 512 KB SRAM + 128 KB zero-wait-state Flash 01: On-chip 448 KB SRAM + 192 KB zero-wait-state Flash 10, 11: On-chip 384 KB SRAM + 256 KB zero-wait-state Flash Note: Bit1~0 can be changed only when the security library is disabled.
	Bit 7:2	Reserved
1024 K and above Flash	Bit 2:0	000: On-chip 512 KB SRAM + 128 KB zero-wait-state Flash 001: On-chip 448 KB SRAM + 192 KB zero-wait-state Flash 010: On-chip 384 KB SRAM + 256 KB zero-wait-state Flash 011: On-chip 320 KB SRAM + 320 KB zero-wait-state Flash 100: On-chip 256 KB SRAM + 384 KB zero-wait-state Flash 101: On-chip 192 KB SRAM + 448 KB zero-wait-state Flash 110, 111: On-chip 128 KB SRAM + 512 KB zero-wait-state Flash Note: Bit 2~0 can be changed only when the security library is disabled.
	Bit 7:3	Reserved

The core can read instruction codes stored in zero-wait-state Flash without any delay, and there is no need to insert a wait state.

Taking AT32F435ZMT7(4032 K Flash) as an example, the following sections focus on how to extend SRAM from 384 KB to 512 KB. For more details about SRAM extension, refer to *AN0026_Extending_SRAM_in_User's_Program* fromARTERY's official website→SUPPORT→AP Note→AN0026.

1.2.3.1 Artery ICP Programmer (BOOT0=0,BOOT1=0)

Connect AT-Link-EZ /AT-Link /J-Link to MCU → Target → User system data → EOPB0, select 512 KB (complete related settings if any) → Apply to device.

Figure 17. ICP Programmer – User system data

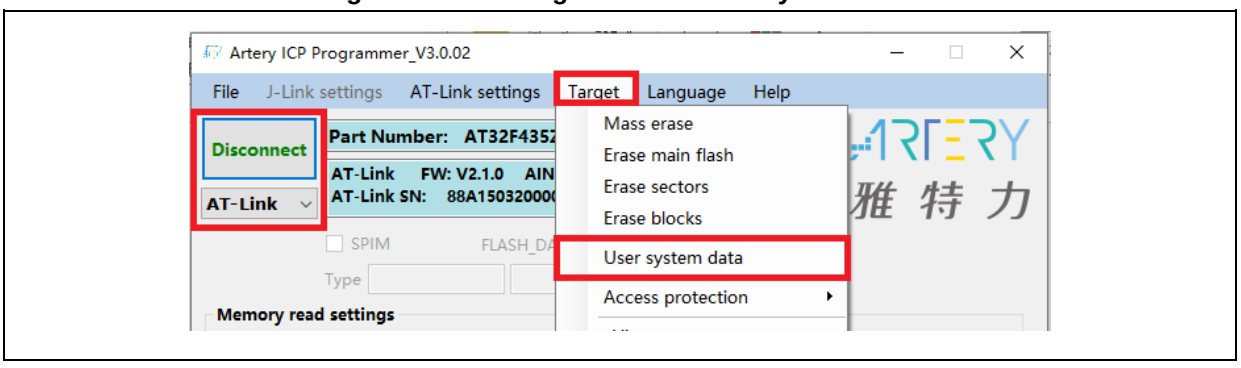
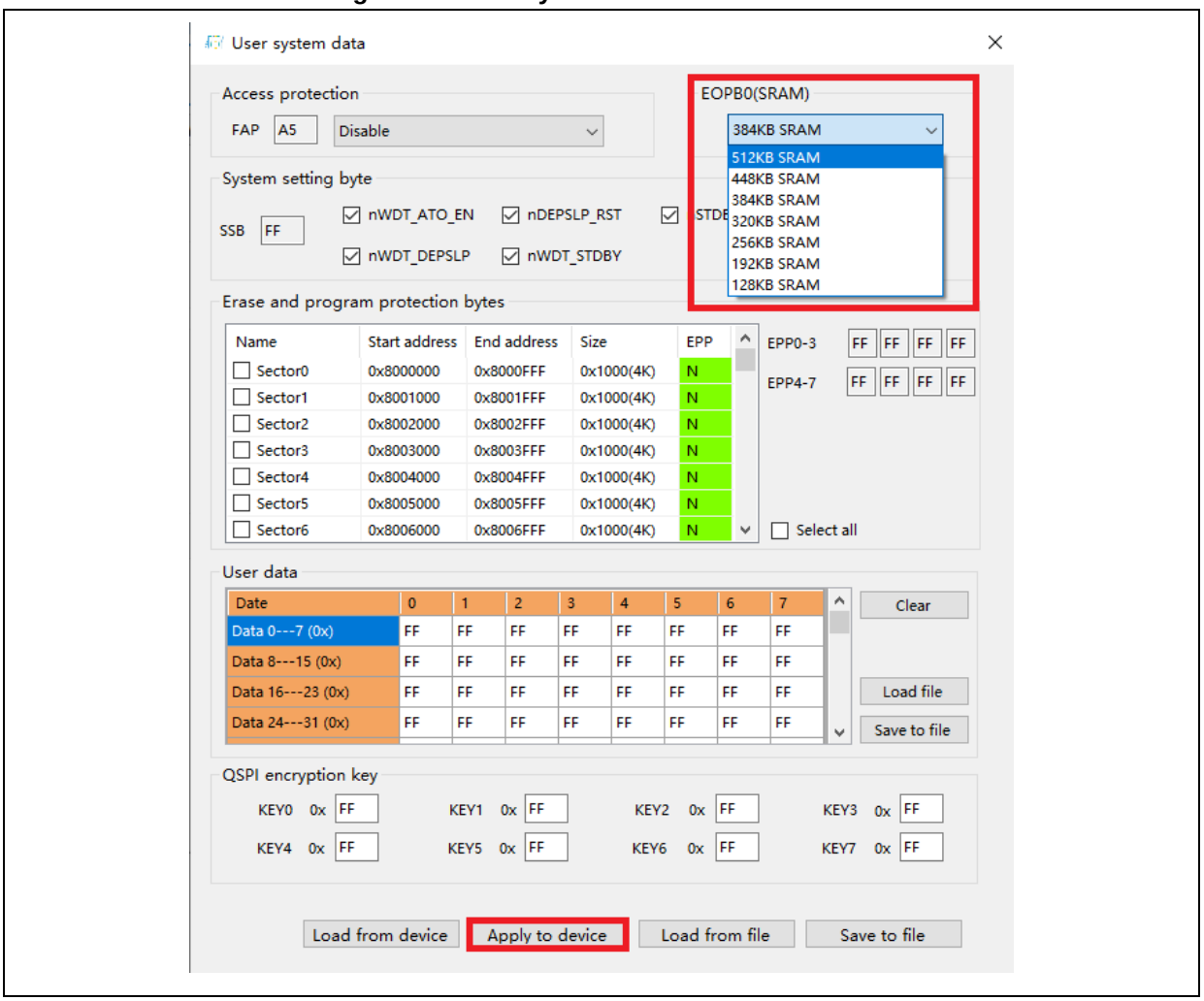


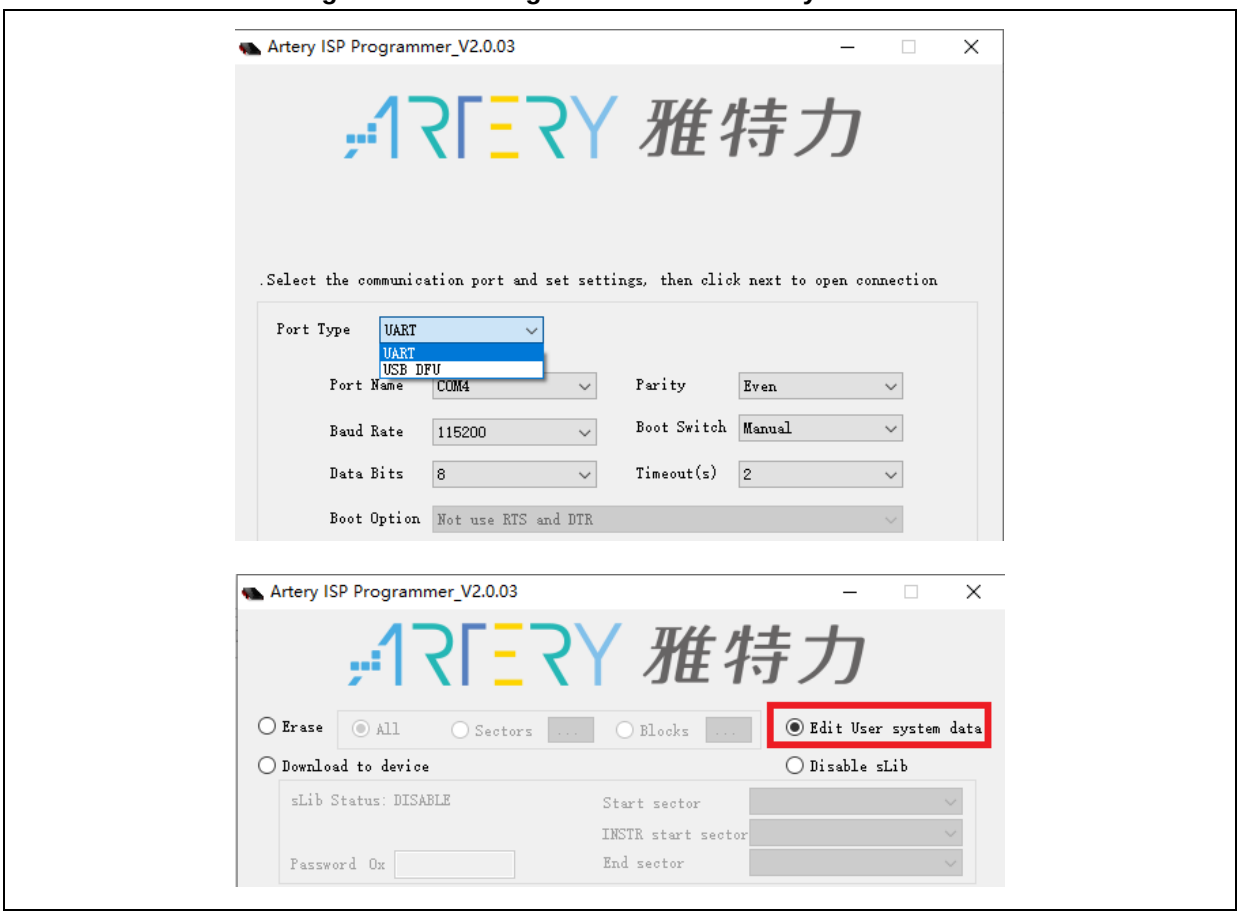
Figure 18. User system data - set SRAM size



1.2.3.2 Artery ISP Programmer (BOOT0=1,BOOT1=0)

Connect UART or USB to MCU (BOOT0=1, BOOT1=0) → click “Next” to enter the following interface → Edit User system data → Next → EOPB0, select 512 KB (complete related settings if any) → Apply to device.

Figure 19. ISP Programmer - Edit User system data



1.2.3.3 Modification in IAP

Users can also modify SRAM size in Bootloader program (IAP). Note that SRAM size in the compiler needs to be the same as the modified SRAM size (see AN0026).

The SRAM is loaded when running the startup file. If the SRAM used by application is larger than 384 KB, users need to extend the SRAM (extend to 512 KB, for example) in IAP. The address of EOPB0 bit in user system data area is 0x1FFFC010, and the code is as follow.

Figure 20. Define Extend_SRAM(void) to modify SRAM size

```
#define   SRAM_384K   0x2
#define   SRAM_512K   0x0
static uint32_t f_eopb0;
f_eopb0=*(uint32_t*)(0x1FFFC010);
void Extend_SRAM(void)
{
    if((f_eopb0 & 0x07) == SRAM_384K) // check if RAM has been set to 384K, if yes, change EOPB0
    {
      /* Unlock User System Data Program Erase controller */
      flash_unlock();
```

```
            /* Erase User System Data */
            flash_user_system_data_erase();
            /* Change SRAM size to 512KB */
            flash_user_system_data_program(0x1FFFC010, SRAM_512K);
            /* Set other user system data…*/
            flash_lock();
            nvic_system_reset();
        }
    }
```

Erase user system data area before changing user system data. If other options in user system data area are set, read and then erase these settings, and then re-write together with SRAM size.

1.2.3.4 Modification in startup file

The SRAM is loaded when running the startup file. If the program does not have IAP and the SRAM used by application is larger than 384 KB, it will lead to a load failure and enter hardfault, causing an application failure. In this case, users can extend SRAM to 512 KB before loading SRAM in the startup file.

The code in bold font is added to the Keil startup file.

Figure 21. Modify SRAM size in Keil startup file

```
; Reset handler
Reset_Handler    PROC
                 EXPORT  Reset_Handler                    [WEAK]
                 IMPORT  __main
                 IMPORT  SystemInit

                 IMPORT   Extend_SRAM
                 MOV32    R0, #0x20001000
                 MOV      SP, R0
                 LDR      R0, =Extend_SRAM
                 BLX      R0
                 MOV32    R0, #0x08000000
                 LDR      SP, [R0]

                 LDR      R0, =SystemInit
                 BLX      R0
                 LDR      R0, =__main
                 BX       R0
                 ENDP
```

The code in bold font is added to the IAR startup file.

Figure 22. Modify SRAM size in IAR startup file

```
; Default interrupt handlers.
        THUMB

        PUBWEAK Reset_Handler
        SECTION .text:CODE:REORDER:NOROOT(2)

        EXTERN    Extend_SRAM
Reset_Handler
        MOV32     R0,#0x20001000
        MOV       SP,R0
        LDR       R0,=Extend_SRAM
        BLX       R0
        MOV32     R0,#0x08000000
        LDR       SP,[R0]

        LDR       R0, =SystemInit
        BLX       R0
        LDR       R0, =__iar_program_start
        BX        R0
```

After completing the above configuration, add declaration and define the “Extend_SRAM” function in application (refer to Section 1.2.3.3), and define the “Extend_SRAM(void)” to modify the SRAM size.

It is not recommended to use APP to modify SRAM size. If the SRAM used by APP is greater than the modified SRAM, the program will enter Hardfault.

1.2.4 Encryption (access protection, erase/program protection)

1.2.4.1 Access protection

Access protection is commonly referred to as “encryption” and applies to the entire Flash memory. Once the Flash access protection is enabled, the internal Flash memory can only be read through normal execution of the program, instead of through JTAG or SWD. Using ISP or ICP tool to unlock access protection will trigger erase operation to the Flash memory

Users can use ICP/ISP programmer to enable or disable access protection.

- Artery ICP Programmer (BOOT0=0,BOOT1=0)

  Enable access protection: Target--Access protection--ENABLE.

  Disable access protection: Target—Access protection--DISABLE.

Figure 23. ISP programmer – enable access protection

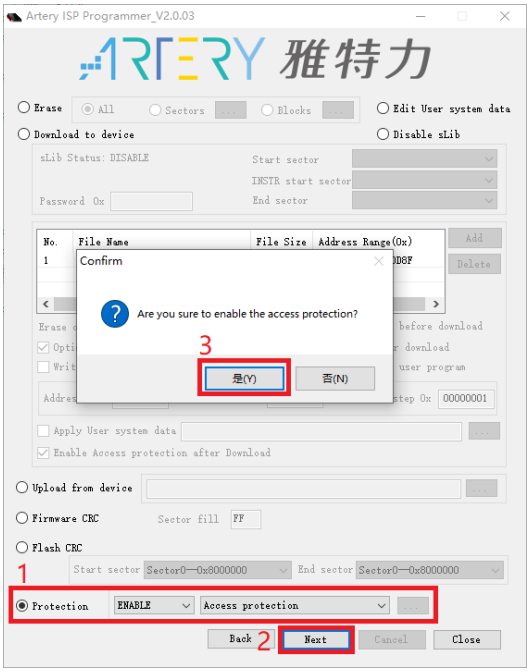
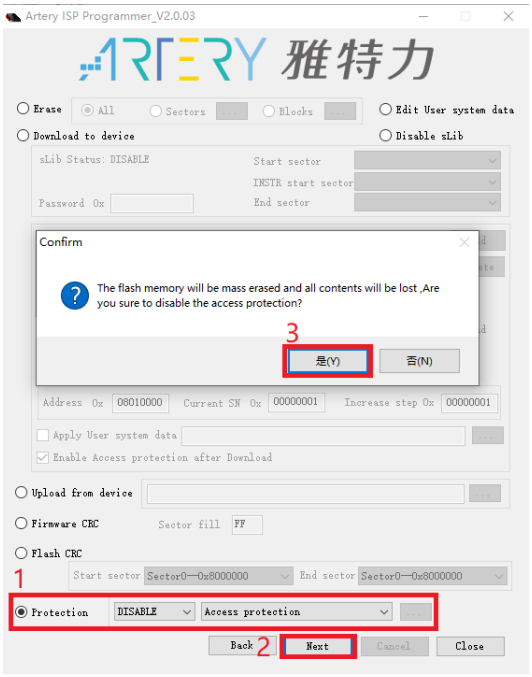


Figure 24. ISP programmer – disable access protection



- Artery ISP Programmer (BOOT0=1, BOOT1=0)

  Enable access protection: Are you sure to enable the access protection?--Yes.

  Disable access protection: Are you sure to disable the access protection?--Yes.

- Artery ISP Multi-Port Programmer (BOOT0=1,B OOT1=0)

  Enable access protection: Are you sure to enable the access protection?--Yes.

  Disable access protection: Are you sure to disable the access protection?--Yes.

Note: Access protection, after being enabled, cannot be unlocked through erase operation.

1.2.4.2 Erase/program protection

Write protection applies to the entire Flash memory or to part of Flash area. Once the Flash write protection is enabled, the internal Flash cannot be written.

Users can use ICP/ISP programmer to enable or disable erase/program protection.

■ Artery ICP Programmer (BOOT0=0, BOOT1=0)

Enable erase/program protection: Target – User system data – tick the page to be erase/program-protected – Apply to device.

Disable erase/program protection: Target – User system data – untick the page to be erase/program-protected – Apply to device.

Figure 25. ICP Programmer – enable erase/program protection

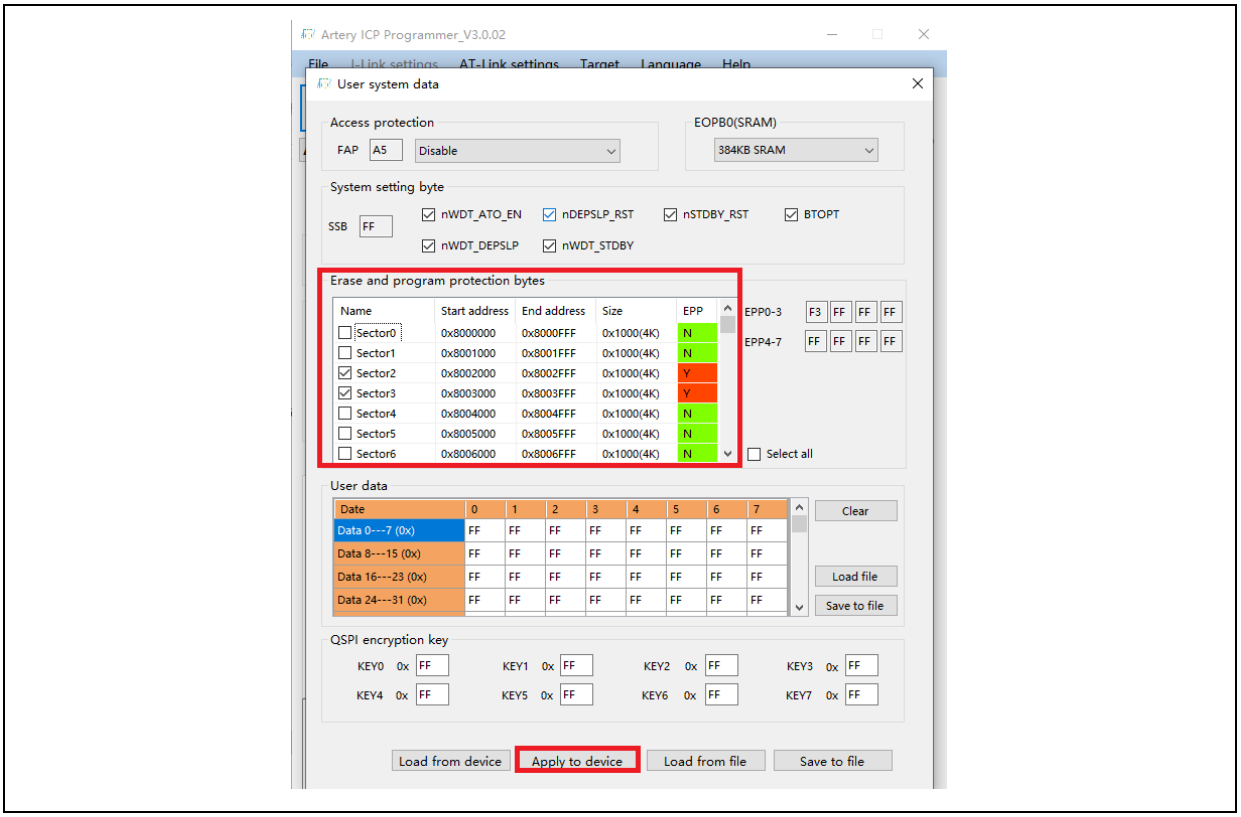
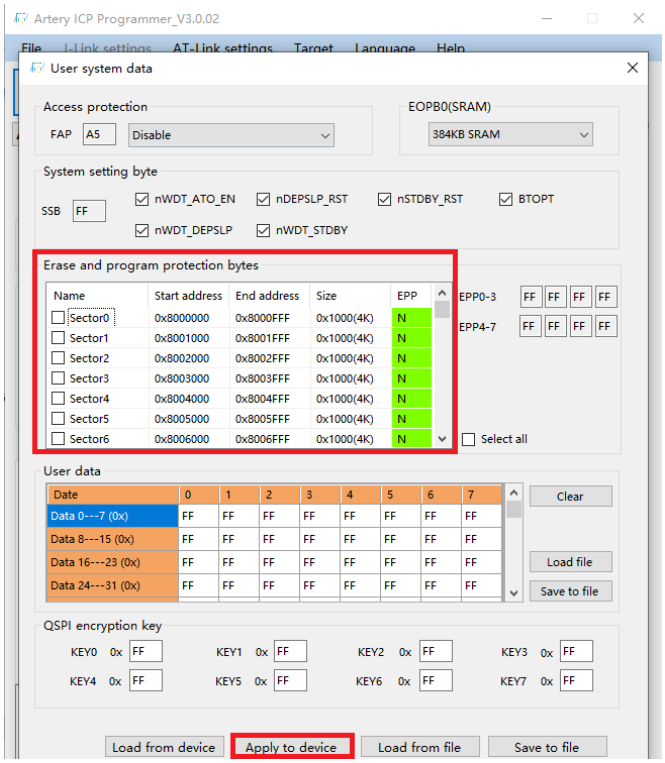


Figure 26. ICP Programmer – disable erase/program protection



- Artery ISP Programmer (BOOT0=1, BOOT1=0)

  Enable erase/program protection: Are you sure to enable erase/program protection?--Yes.

  Disable erase/program protection: Are you sure to disable erase/program protection?--Yes.

- Artery ISP Multi-Port Programmer (BOOT0=1, BOOT1=0)

  Enable erase/program protection: Are you sure to enable erase/program protection?--Yes.

  Disable erase/program protection: Are you sure to disable erase/program protection?--Yes.

Note: The erase/program protection, after being enabled, cannot be unlocked through erase operation.

1.2.5 How to distinguish AT32 MCU from other MCUs

■ Read Cortex-M series CPU ID number to determine whether it is based M0, M3 or M4 core.

Figure 27. Read Cortex ID

```
cortex_id = *(uint32_t *)0xE000ED00;// read Cortex ID
if((cortex_id == 0x410FC240) || (cortex_id == 0x410FC241))
{
        printf("This chip is Cortex-M4F.\r\n");
}
else
{
        printf("This chip is Other Device.\r\n");
}
```

■ **Read UID and PID**

Figure 28. Read UID and PID

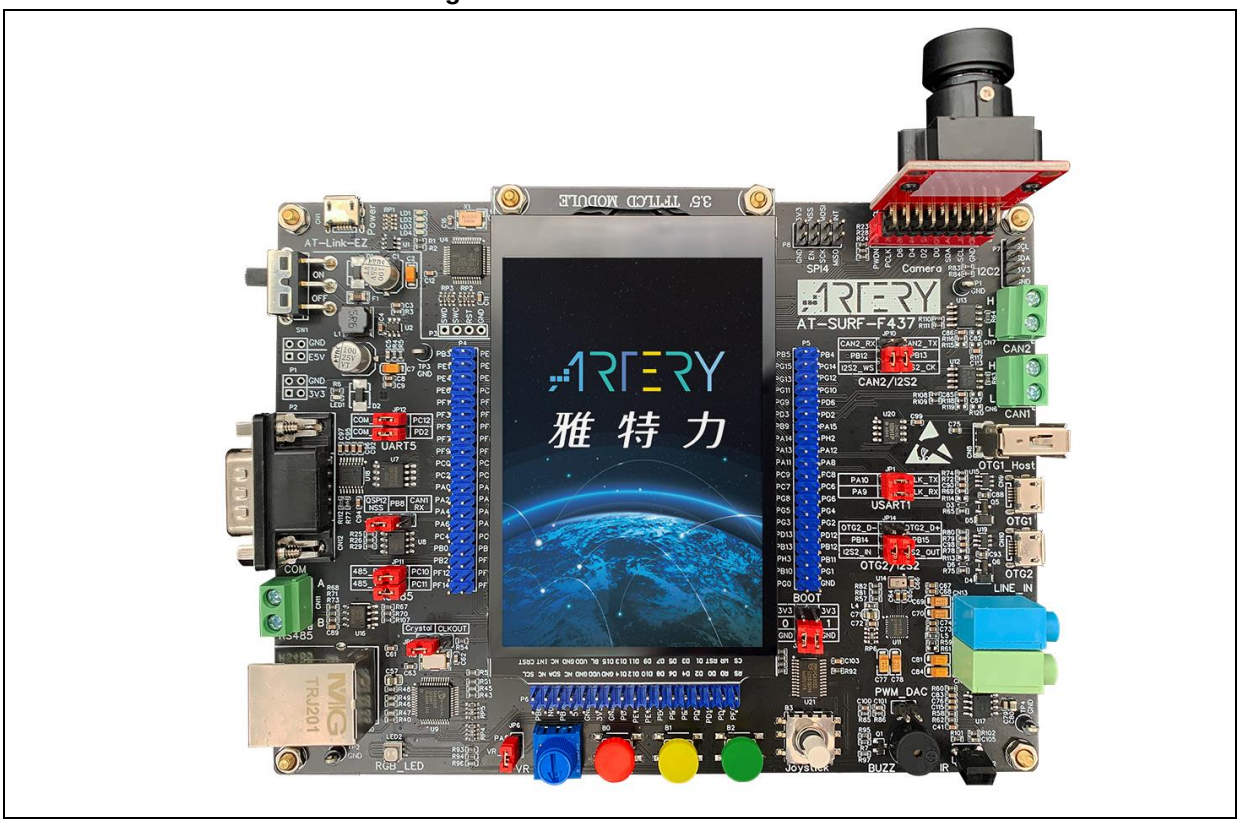
```
/* Get AT32 MCU PID/UID base address */
#define DEVICE_ID_ADDR1 0x1FFFF7F3          // define MCU device ID and UID base address
#define DEVICE_ID_ADDR2 0xE0042000          // define MCU device ID and PID base address

/* store ID */
uint8_t     ID[5] = {0};

/* AT32F435 MCU type table */
const uint64_t AT32_MCU_ID_TABLE[] =
{
      0x0000000D70084540,          //AT32F435ZMT7     4032KB    LQFP144
      0x0000000D7008454F,          //AT32F437ZMT7     4032KB    LQFP144
      …
};

  /* Get UID/PID */
ID[0] = *(int*)DEVICE_ID_ADDR1;
ID[1] = *(int*)(DEVICE_ID_ADDR2+3);
ID[2] = *(int*)(DEVICE_ID_ADDR2+2);
ID[3] = *(int*)(DEVICE_ID_ADDR2+1);
ID[4] = *(int*)(DEVICE_ID_ADDR2+0);

/* combine UID/PID */
  AT_device_id =
((uint64_t)ID[0]<<32)|((uint64_t)ID[1]<<24)|((uint64_t)ID[2]<<16)|((uint64_t)ID[3]<<8)|((uint64_t)ID[4]<<0);

/* Identify AT32 MCU */
for(i=0;i<sizeof(AT32_MCU_ID_TABLE)/sizeof(AT32_MCU_ID_TABLE[0]);i++)
{
    if(AT_device_id == AT32_MCU_ID_TABLE[i])
    {
        printf("This chip is AT32F4xx.\r\n");
    }
      else
    {
        printf("This chip is Other Device.\r\n");
    }
}
```

Note: AT32F4xx MCU contains several ID codes. By organizing the obtained ID information into a 64-bit data, it is possible for users to determine which MCU series is being used. For details, refer to the “Debug” section of the corresponding reference manual and *AN0016_Recognize_AT32_MCU* from ARTERY’s official website→SUPPORT→AP Note→AN0016.

1.2.6 AT32F435/AT32F437 advanced functions

ARTERY offers AT-SURF-F437 board that has most of the advanced functions of AT32F437, and provides practical programs (stored in “\project\at_surf_f437 \examples” in BSP). For details, please refer to *AN0049_AT_SURF_F437_Board_Application_Note* from ARTERY’s official website→SUPPORT→AP Note→AN0049.

Figure 29. AT-SURF-F437 board



Note: For more information on how to get better AT32F435_437 operating performance, please refer to "AN0004_Performance_Optimization" and "AN0092_AT32F435_437_Performance_Improve" available from ARTERY's official website→SUPPORT→AP Note→AN0004/AN0092.

2 FAQs about download and compiling

2.1 Program enters Hard Fault Handler

- The SRAM used by the program is outside the SRAM threshold programmed in user system data area.

  Refer to 1.2.3 and use ICP/ISP tools or a third-party programmer to extend SRAM.

- The “single precision” function is enabled in Keil or IAR, but the M4 core FPU register is not enabled in the code. In this case, users need to enable FPU feature in the code.

Figure 30. Add code to enable FPU

```
void SystemInit (void)
{
  /* Enable FPU*/
#if defined (__FPU_USED) && (__FPU_USED == 1U)
SCB->CPACR |= ((3U << 10U * 2U) |           /* set CP10 Full Access */
               (3U << 11U * 2U)  );         /* set CP11 Full Access */
 #endif
```

- Access data outside its boundary limit

  Locate where the program exceeds the boundary, and move it to normal data area.

- System clock is set out of specification.

2.2 J-Linkcannot find IC

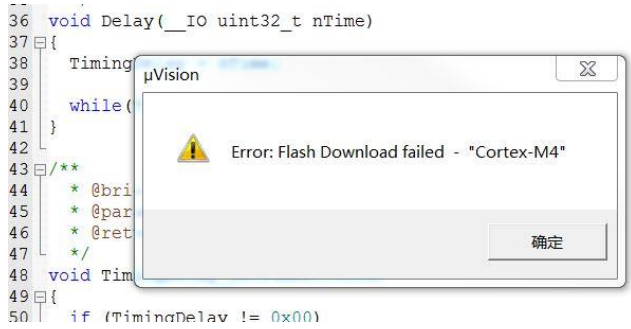
- Please refer to *FAQ0008_ J-Link cannot find IC* (ARTERY’s official website →SUPPORT→FAQ→FAQ0008)
- Please refer to *FAQ0132_Add Artery MCU to J-Link* (ARTERY’s official website →SUPPORT→FAQ→FAQ0132)

2.3 Errors during download

2.3.1 Flash Download failed–“Cortex-M4”

A warning message pops up during Keil debugging or downloading:

Figure 31. Error: Flash Download failed–“Cortex- 4”



The warning message pops up in one of the following conditions:

- Access protection is enabled. Disable MCU access protection before download.
- An incorrect Flash algorithm file is selected or Flash algorithm file is not loaded. Select and add the correct Flash algorithm to “Flash Download”.
- BOOT0 and BOOT1 are incorrect. Set BOOT0=0 and BOOT1=0 to boot MCU from the main Flash memory.
- J-Link driver version is incorrect. Versions 6.20C and above are recommended.
- The JTAG/SWD PIN is disabled. Refer to Section 2.2.5 for solution.

2.3.2 No Debug Unit Device found

- Download interface is being occupied. For example, ICP is being connected to a target device.
- JTAG/SWD connection error or it is not connected.

2.3.3 RDDI-DAP Error

- The compiler optimization level is too high. For example, Keil AC6 optimization level is the default “-Oz”, which should be changed to “-O0/-O1”.
- The JTAG/SWD PIN is disabled. Refer to Section 2.2.5 for solution.

2.3.4 ISP serial interface gets stuck during download

When the ISP serial interface is used for download, it may get stuck so that it cannot be released. It is recommended to:

- Check if the power supply is stable.
- Use a better USB-to-serial interface tool, such as CH340 chip.

2.3.5 How to resume program download

Users may not be able to download programs in one of the following conditions:

- Disabled JTAG/SWD PIN, so that program download failed and the JTAG/SWD device cannot be found.
- Entered Standby mode, so that program download failed and the JTAG/SWD device cannot be found.

The following solutions are recommended:

- Solution 1: Switch boot mode
  Switch boot mode to Boot[1:0]=01b or Boot[1:0]=11b, and then press “Reset” button to resume download (note to return to Boot[1:0]=00b after download resumes). This method also applies to ISP download.
- Solution 2: Use ICP tool to add AT-Link
  The AT-Link is specially designed for AT32 MCUs; therefore, it is possible to resume download by adding AT-Link through ICP tool.

3 Security Library (sLib)

3.1 Introduction

As more and more MCU applications require complex algorithms and middleware solutions, it has become an important issue that how to protect IP-Codes (such as core algorithms) developed by software solution providers.

In response to this demand, the AT32F435/437 series is designed with a security library (sLib) to protect important IP-Codes against being changed or read by the end user program.

3.2 Application principles

- Security library (sLib) is a defined area protected by a code in the main memory. Software solution providers store core algorithms in sLib for protection. The rest of the area can be used for secondary development by end users.
- Security library includes the read-only area (SLIB_READ_ONLY) and instruction area ((SLIB_INSTRUCTION), and it can be partially or completely used as the read-only area or instruction area.
- Data of the read-only area (SLIB_READ_ONLY) can be read by I-Code and D-Code buses but cannot be written.
- Program codes in the instruction area (SLIB_INSTRUCTION) can only be fetched by MCU through I-Code bus (only executable), and cannot be read by reading access through D-Code bus (including ISP/ICP/debug mode or boot from internal RAM), for accessing SLIB_INSTRUCTION by reading data operation will return all 0xFF.
- Program codes and data in security library cannot be erased unless the correct code is keyed in. If a wrong code is keyed in, in an attempt of writing or deleting security library code, a warning message will be issued by EPPERR=1 in the FLASH_STS register.
- Mass erase operation to the main Flash memory by end users will not erase the codes and data in security library.
- After sLib is enabled, users can also unlock the sLib protection by writing the previously defined password in the SLIB_PWD_CLR register. After the security library protection is disabled, the MCU will erase the whole main memory, including the sLib. Therefore, the program codes are protected against leakage even if the code defined by the software solution provider is leaked.

3.3 How to use sLib

For more details, please refer to "*AN0081_AT32F435_437_Security_Library_Application_Note*" from ARTERY's official website→SUPPORT→AP Note→AN0081.

4 Revision history

Table 1. Document revision history

Date	Version	Revision note
2022.04.20	2.0.0	Initial release.
2022.07.08	2.0.1	Optimized descriptions.
2022.10.11	2.0.2	Updated the 3rd party programming tools, and added descriptions to the development environment and file path.
2022.10.21	2.0.3	Optimized description of UID and PID.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers understand and agree that purchasers are solely responsible for the selection and use of Artery's products and services.

Artery's products and services are provided "AS IS" and Artery provides no warranties express, implied or statutory, including, without limitation, any implied warranties of merchantability, satisfactory quality, non-infringement, or fitness for a particular purpose with respect to the Artery's products and services.

Notwithstanding anything to the contrary, purchasers acquires no right, title or interest in any Artery's products and services or any intellectual property rights embodied therein. In no event shall Artery's products and services provided be construed as (a) granting purchasers, expressly or by implication, estoppel or otherwise, a license to use third party's products and services; or (b) licensing the third parties' intellectual property rights; or (c) warranting the third party's products and services and its intellectual property rights.

Purchasers hereby agrees that Artery's products are not authorized for use as, and purchasers shall not integrate, promote, sell or otherwise transfer any Artery's product to any customer or end user for use as critical components in (a) any medical, life saving or life support device or system, or (b) any safety device or system in any automotive application and mechanism (including but not limited to automotive brake or airbag systems), or (c) any nuclear facilities, or (d) any air traffic control device, application or system, or (e) any weapons device, application or system, or (f) any other device, application or system where it is reasonably foreseeable that failure of the Artery's products as used in such device, application or system would lead to death, bodily injury or catastrophic property damage.

Any inconsistency of the sold ARTERY products with the statement and/or technical features specification described in this document will immediately cause the invalidity of any warranty granted by ARTERY products or services stated in this document by ARTERY, and ARTERY disclaims any responsibility in any form.