AN0024
Application Note

Getting Started with AT32F421

Introduction

This application note is written to help users with rapid project development using AT32F421xx.

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	AT32F421xx

Contents

1 Preliminary environment requirements 5

1.1 Build AT32 development environment 5

1.1.1 Debug tools and evaluation board 5

1.1.2 Programming tools and software 5

1.1.3 AT32 development environment 6

1.1.4 How to replace SXX 11

1.2 AT32F421 chip enhanced functions 12

1.2.1 Prefetch buffer 12

1.2.2 PLL clock settings 13

1.2.3 Encryption mode 15

1.2.4 Set boot memory as main Flash memory extension area 18

1.2.5 Recognize AT32 MCU in program 21

2 FAQs in downloading/compiling 23

2.1 Hard Fault Handler 23

2.2 J-Link cannot find IC 23

2.3 Errors in program downloading 23

2.3.1 Flash Download failed–“Cortex-M4” 23

2.3.2 No Debug Unit Device found 24

2.3.3 RDDI-DAP Error 24

2.3.4 ISP serial port gets stuck in downloading 24

2.3.5 Resume download 24

3 Security Library (sLib) 25

3.1 Introduction 25

3.2 Principles of application 25

3.3 Security library application 25

4 Revision history 26

List of Tables

Table 1. Document revision history... 26

List of Figures

Figure 1. AT-START-F421 evaluation board with AT-Link-EZ........5
Figure 2. AT-START-F421 evaluation board package........5
Figure 3. ICP/ISP/AT-Link-Family........6
Figure 4. BSP package........6
Figure 5. Keil_v5 templates........7
Figure 6. Pack download........8
Figure 7. Set up ArteryTek.AT32F421 _DFP........8
Figure 8. Set up Keil4_AT32MCU_AddOn........8
Figure 9. Pack Installer icon in Keil........9
Figure 10. Set up IAR_AT32MCU_AddOn........9
Figure 11. Keil Debug option........10
Figure 12. Keil Debug Settings........10
Figure 13. Keil Utilities........10
Figure 14. IAR Debug option........11
Figure 15. IAR CMSIS-DAP option........11
Figure 16. Wait states of Flash performance select register (FLASH_PSR)........12
Figure 17. System clock configuration function “system_clock_config”........12
Figure 18. AT32F421 118 MHz PLL clock configuration........13
Figure 19. SXX PLL auto step-by-step switch configurations........14
Figure 20. AT32 PLL auto step-by-step switch configurations........14
Figure 21. Enable/disable access protection in ICP Programmer........15
Figure 22. Enable access protection in ISP Programmer........16
Figure 23. Disable access protection in ISP Programmer........16
Figure 24. Enable erase and program protection in ICP Programmer........17
Figure 25. Disable erase and program protection in ICP Programmer........18
Figure 26. Set AP mode in ICP Programmer........19
Figure 27. AP mode enabling in ICP Programmer........19
Figure 28. Offline config settings in ICP Programmer........20
Figure 29. AT-Link project file settings........21
Figure 30. AT-Link offline download status........21
Figure 31. Read Cortex ID........22
Figure 32. Read PID and UID........22
Figure 33. Flash Download failed–“Cortex- M4” pops up in downloading........23

1 Preliminary environment requirements

Download development environment

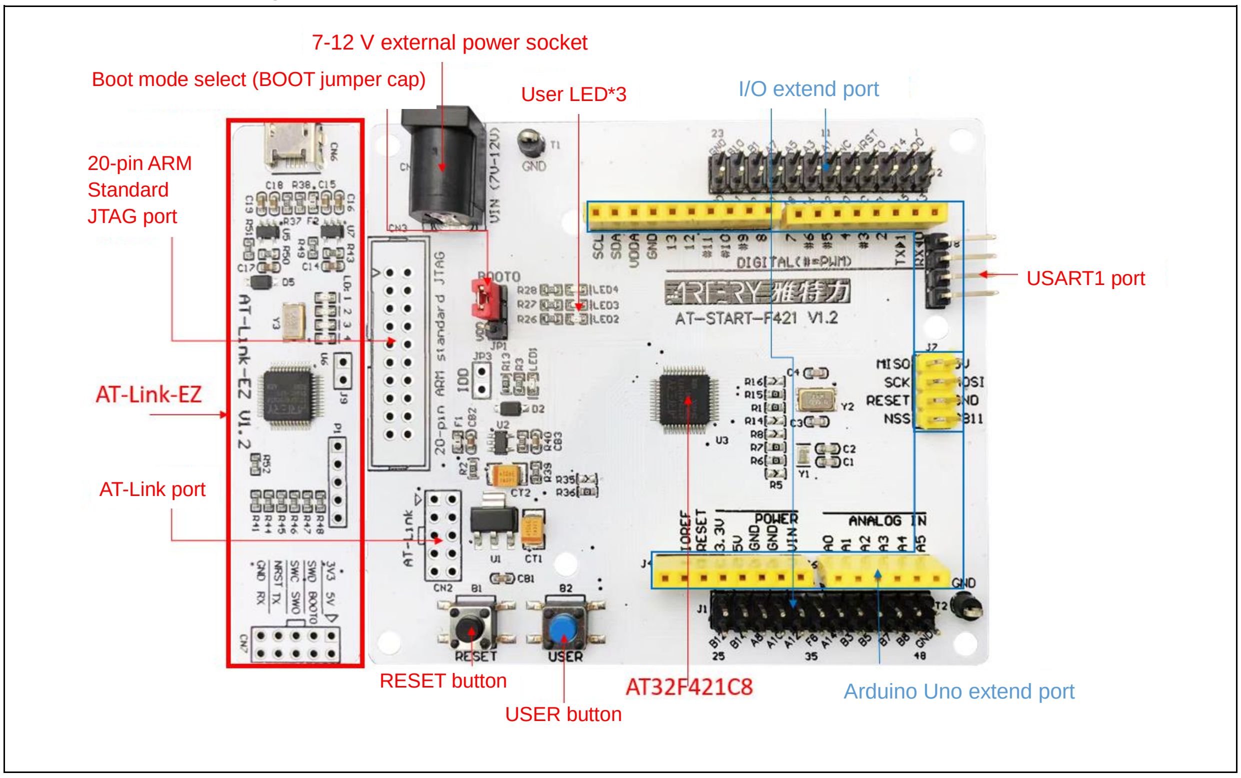
■ ARTERY's official website

1.1 Build AT32 development environment

1.1.1 Debug tools and evaluation board

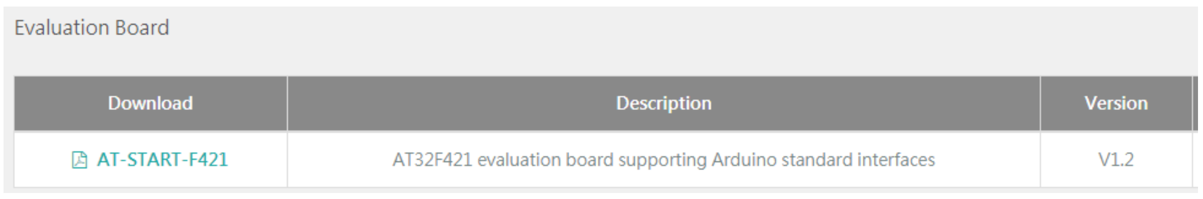
The AT32F421 supports AT-Link/J-Link, and the evaluation board has an AT-Link-EZ debug tool (as shown in the red box in Figure 1 below), which can be disassembled and used with other circuit boards, supporting IDE online debugging, online programming and USB-to-serial port.

Figure 1. AT-START-F421 evaluation board with AT-Link-EZ



Note: For details about resources for AT-START evaluation board, please refer to UM_AT_START_F421_Vx.x. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4xx; download and unzip Evaluation Board package, and get the "\AT_START_F421_Vx.x\03_Documents".

Figure 2. AT-START-F421 evaluation board package



Evaluation Board

Download	Description	Version
AT-START-F421	AT32F421 evaluation board supporting Arduino standard interfaces	V1.2

1.1.2 Programming tools and software

■ AT programming tools and software: AT-Link / AT-Link+ /AT-Link-Pro / AT-Link-ISO /AT-Link-

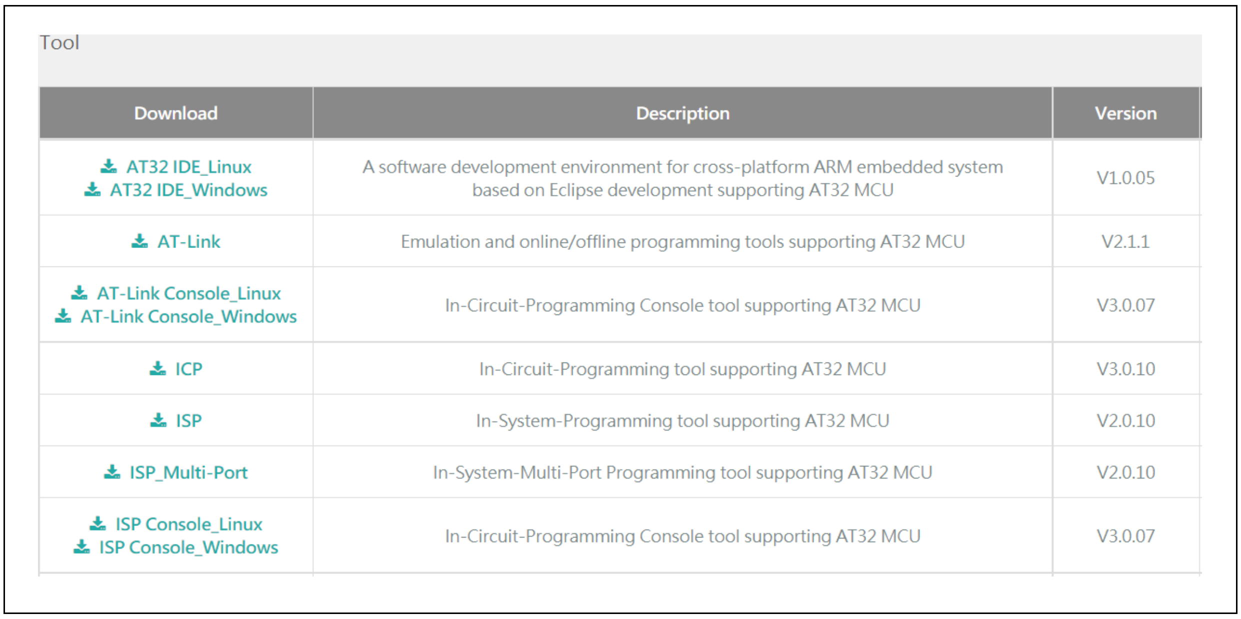
EZ, ICP/ISP.

- 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→Value line→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

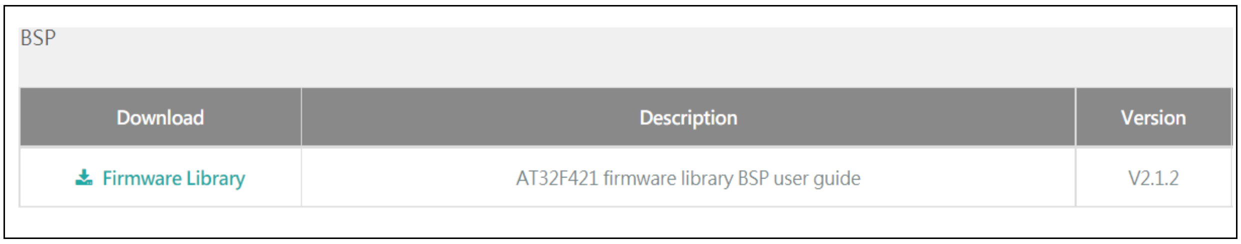
Download	Description	Version
AT32 IDE_Linux AT32 IDE_Windows	A software development environment for cross-platform ARM embedded system based on Eclipse development supporting AT32 MCU	V1.0.05
AT-Link	Emulation and online/offline programming tools supporting AT32 MCU	V2.1.1
AT-Link Console_Linux AT-Link 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→Value line→AT32F4xx.

Figure 4. BSP package

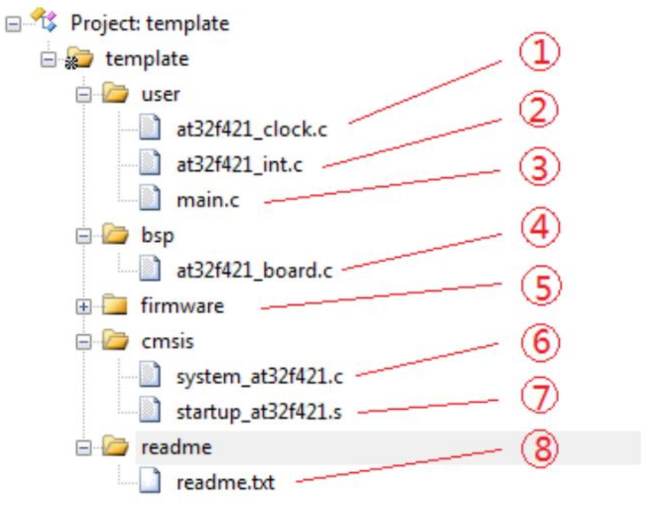


BSP

Download	Description	Version
Firmware Library	AT32F421 firmware library BSP user guide	V2.1.2

Template projects based on Keil_v5/Keil_v4/IAR_6.10/IAR_7.4/IAR_8.2/eclipse_gcc/at32_ide are contained in BSP. Path: AT32F421_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 at32f421_clock.c: clock configuration file, contains default clock frequency and clock path;

② at32f421_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 at32f421_board.c: board-level configuration file, contains settings of AT-START on-board buttons and LEDs;

⑤ at32f421_xx.c under *firmware* file: driver file of on-chip peripherals;

⑥ system_at32f421.c: system initialization file;

⑦ startup_at32f421.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.

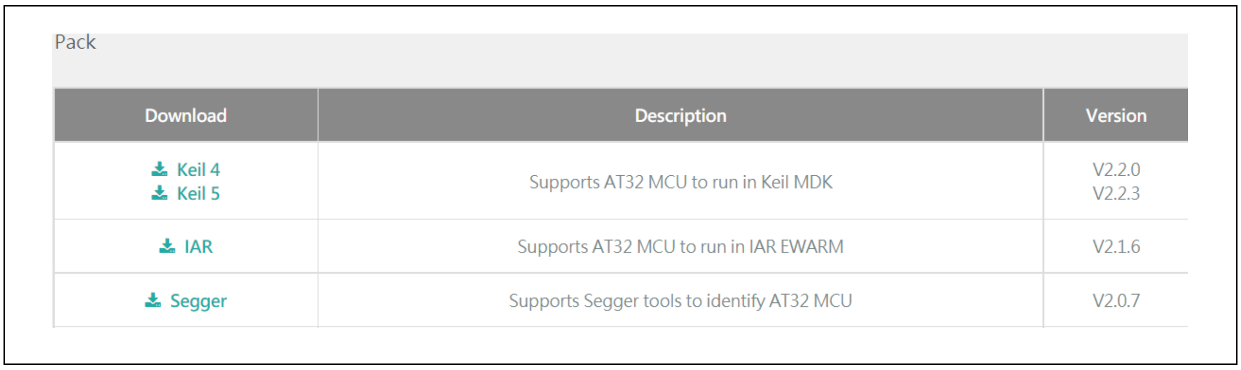
Path: AT32F421_Firmware_Library_V2.x.x\project\at_start_f4xx\examples.

Note: For more details about BSP, please refer to "Section 4 BSP application" of AT32F421 Firmware BSP&Pack User Guide. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4xx; download and unzip, and get the "\AT32F421_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→Value line→AT32F4xx.

Figure 6. Pack download

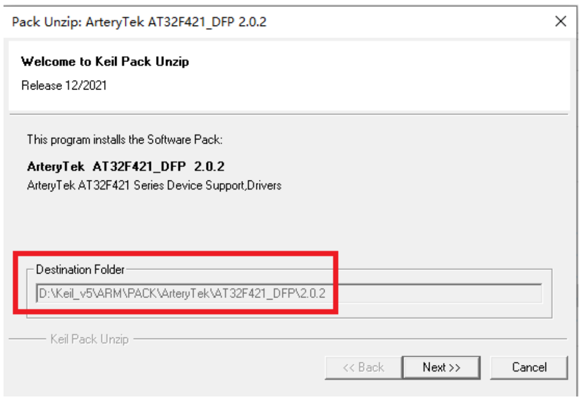


Pack

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.AT32F421_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. Set up ArteryTek.AT32F421 _DFP



Pack Unzip: ArteryTek AT32F421_DFP 2.0.2

Welcome to Keil Pack Unzip

Release 12/2021

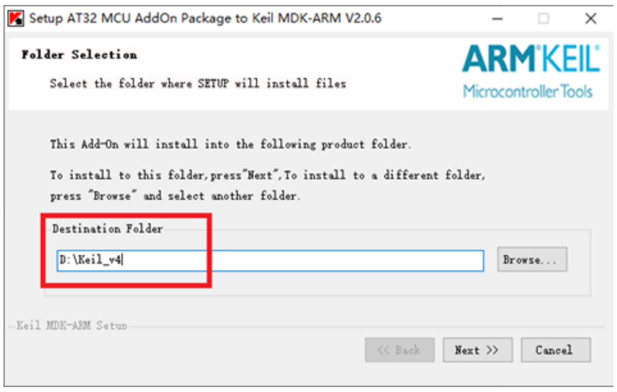
This program installs the Software Pack:

ArteryTek AT32F421_DFP 2.0.2

ArteryTek AT32F421 Series Device Support,Drivers

Destination Folder: D:\Keil_v5\ARM\PACK\ArteryTek\AT32F421_DFP\2.0.2

Figure 8. Set up Keil4_AT32MCU_AddOn



Setup AT32 MCU AddOn Package to Keil MDK-ARM V2.0.6

Folder Selection

Select the folder where SETUP will install files

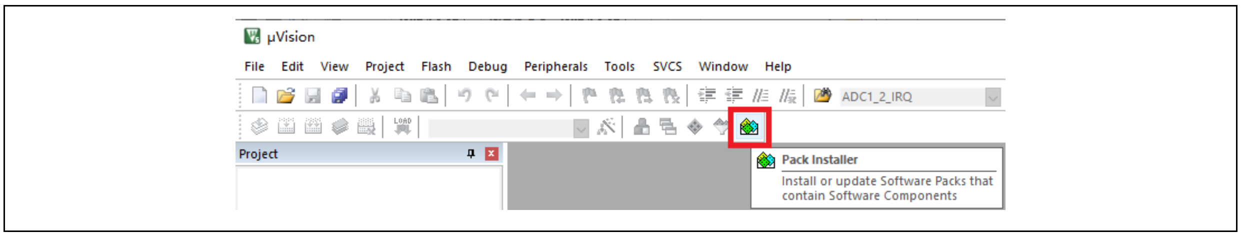
This Add-On will install into the following product folder.

To install to this folder, press"Next",To install to a different folder, press "Browse" and select another folder.

Destination Folder: D:\Keil_v4

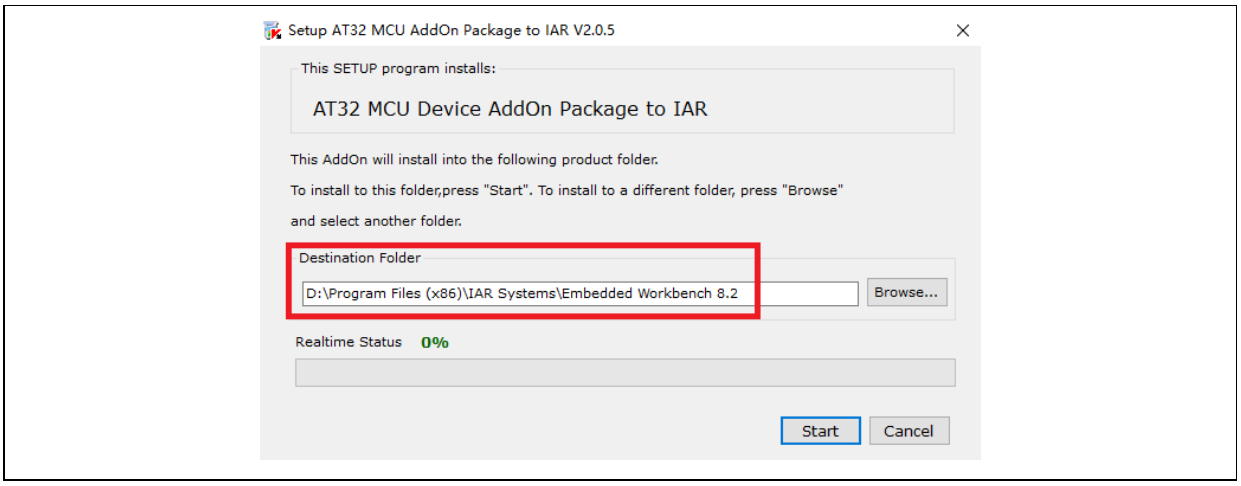
You can also open keil and click on “Pack Installer” icon; then click on the top left “file” and select “import” to import the corresponding pack downloaded from ARTERY's official website.

Figure 9. Pack Installer icon in Keil


µVision
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
ADC1_2_IRQ
Project
Pack Installer
Install or update Software Packs that contain Software Components

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. Set up IAR_AT32MCU_AddOn



Setup AT32 MCU AddOn Package to IAR V2.0.5

This SETUP program installs:

AT32 MCU Device AddOn Package to IAR

This AddOn will install into the following product folder.

To install to this folder,press "Start". To install to a different folder, press "Browse" and select another folder.

Destination Folder

D:\Program Files (x86)\IAR Systems\Embedded Workbench 8.2

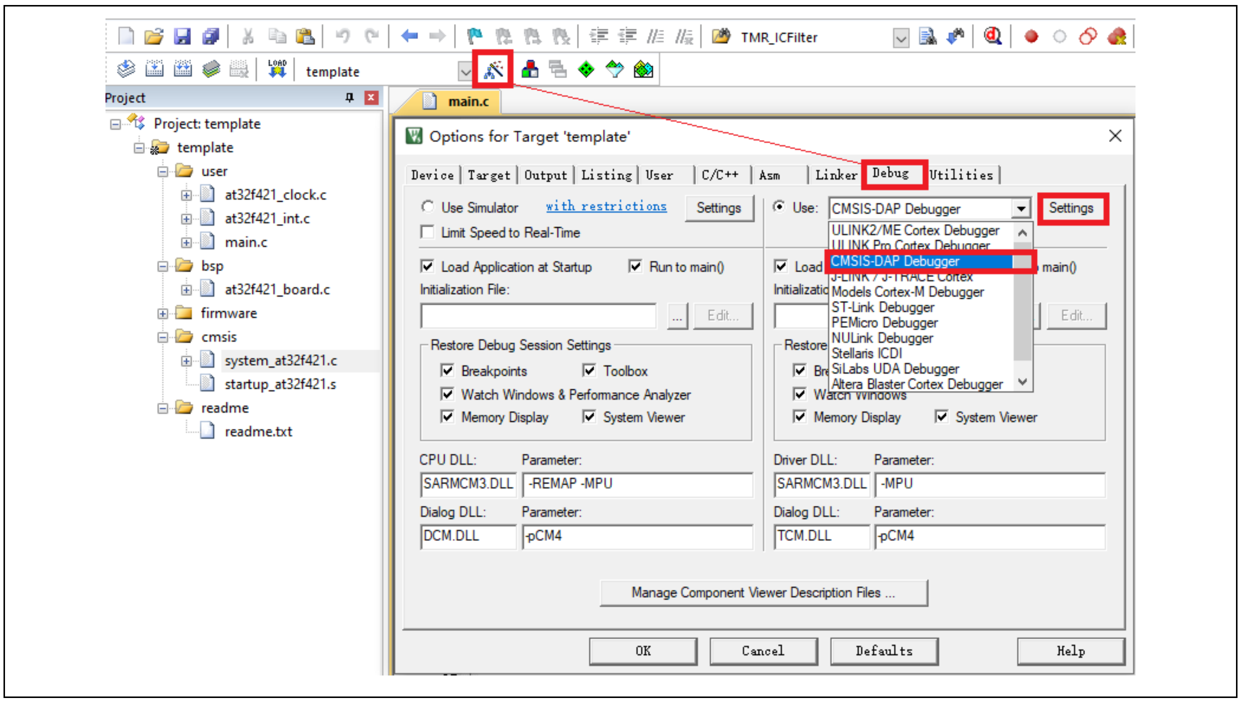
Realtime Status 0%

Note: For more details about Pack setup, please refer to "Section 2 Pack setup" of AT32F421 Firmware BSP&Pack User Guide. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4xx; download and unzip BSP, and get the "\AT32F421_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



Options for Target 'template'

Use Simulator with restrictions Settings
Limit Speed to Real-Time
☑ Load Application at Startup ☑ Run to main()
Initialization File:
Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watch Windows & Performance Analyzer
☑ Memory Display ☑ System Viewer
CPU DLL: SARMCM3.DLL Parameter: -REMAP -MPU
Dialog DLL: DCM.DLL Parameter: -pCM4

● Use: CMSIS-DAP Debugger Settings
ULINK2/ME Cortex Debugger
ULINK Pro Cortex Debugger
CMSIS-DAP Debugger
J-LINK / J-TRACE Cortex
Models Cortex-M Debugger
ST-Link Debugger
PEMicro Debugger
NULink Debugger
Stellaris ICDI
SiLabs UDA Debugger
Altera Blaster Cortex Debugger
Driver DLL: SARMCM3.DLL Parameter: -MPU
Dialog DLL: TCM.DLL Parameter: -pCM4

Manage Component Viewer Description Files ...

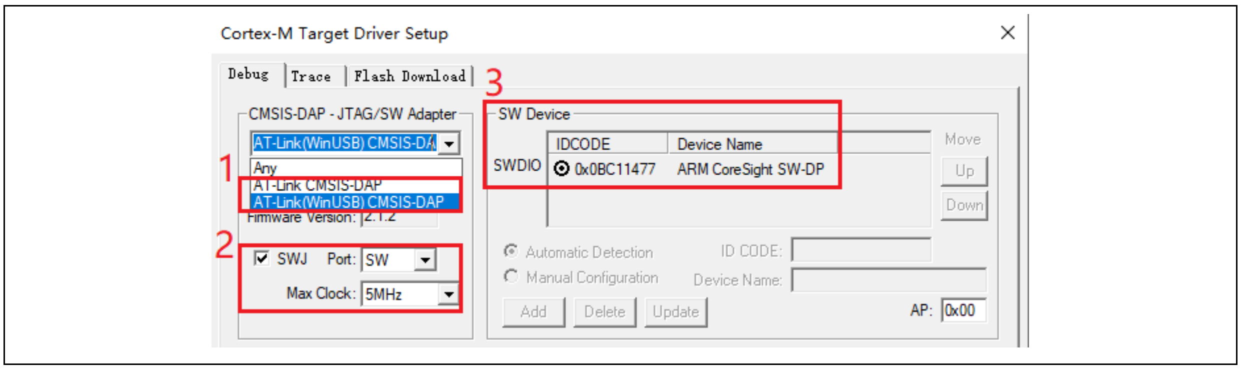
Go to Debug and click on Settings to enter the Cortex-M Target Driver Setup interface.

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



Cortex-M Target Driver Setup

CMSIS-DAP - JTAG/SW Adapter: AT-Link(WinUSB) CMSIS-DAP
Any
AT-Link CMSIS-DAP
AT-Link(WinUSB) CMSIS-DAP
Firmware Version: 2.1.2
☑ SWJ Port: SW
Max Clock: 5MHz

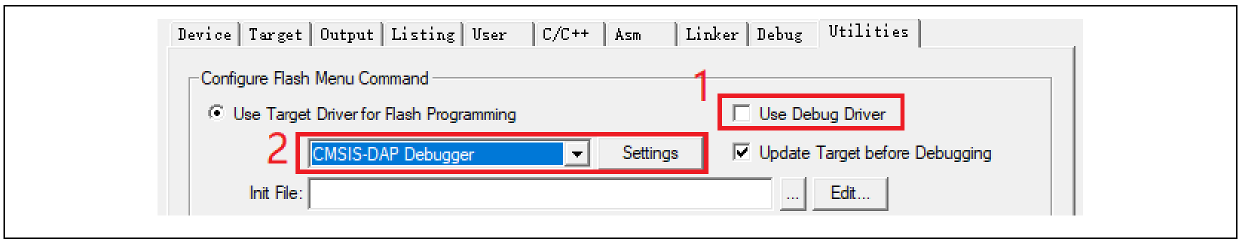
SW Device

	IDCODE	Device Name
SWDIO	0x0BC11477	ARM CoreSight SW-DP

● Automatic Detection ID CODE:
○ Manual Configuration Device Name:
AP: 0x00

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).

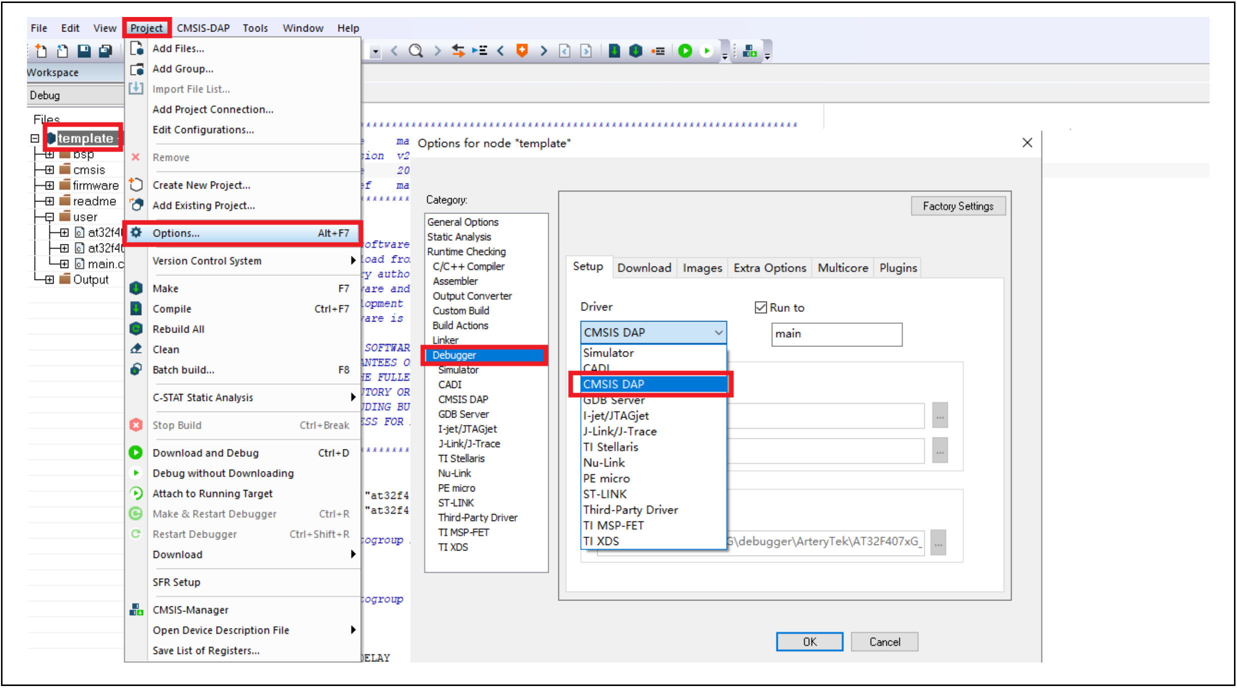
Figure 13. Keil Utilities



Configure Flash Menu Command
● Use Target Driver for Flash Programming ☐ Use Debug Driver
CMSIS-DAP Debugger ☑ Update Target before Debugging
Init File:

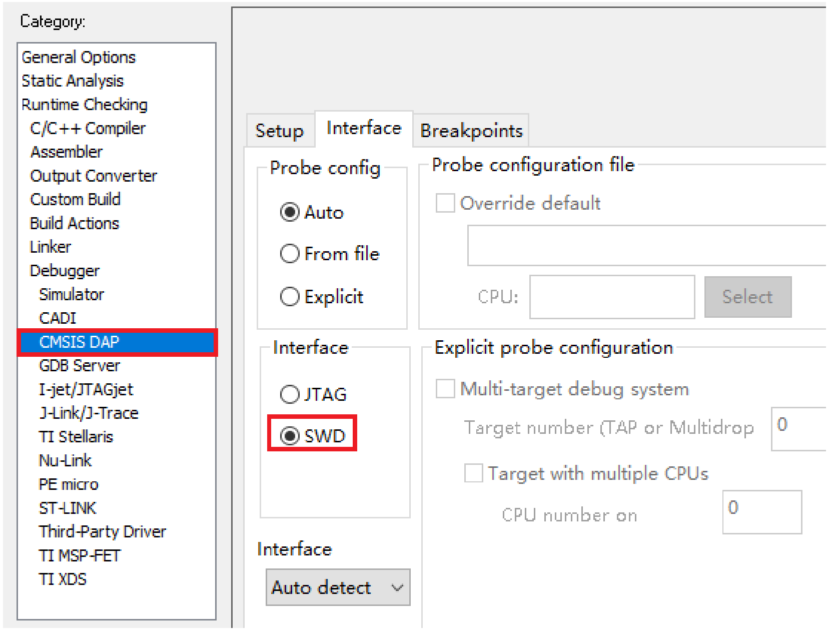
If you want to use AT-Link in IAR, please click on Project and select Options; then select CMSIS-DAP in Debugger option, and select SWD in CMSIS DAP option.

Figure 14. IAR Debug option



Driver: CMSIS DAP
Run to: main
\debugger\ArteryTek\AT32F407xG_

Figure 15. IAR CMSIS-DAP option



Probe config: [✓ Auto] From file Explicit
Interface: JTAG [✓ SWD]
Interface: Auto detect
Target number (TAP or Multidrop: 0
CPU number on: 0

Note: For details about Flash algorithm file, MCU switch and solutions for "J-Link cannot find MCU", please refer to AT32F421 Firmware BSP&Pack User Guide. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4xx; download and unzip BSP, and get the "\AT32F421_Firmware_Library_Vx.x.x\document".

1.1.4 How to replace SXX

- Please refer to *MG0016_ Migrating from SXX32F030 to AT32F421*. Path: ARTERY's official website→PRODUCTS→Value line→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.

Note: Compared with SXX32F0xx, AT32F421 is more flexible to achieve better performance. Please refer to AN0004_Performance_Optimization (ARTERY's official website→ SUPPORT→AP Note→AN0004) to learn how to promote performance of AT32F421.

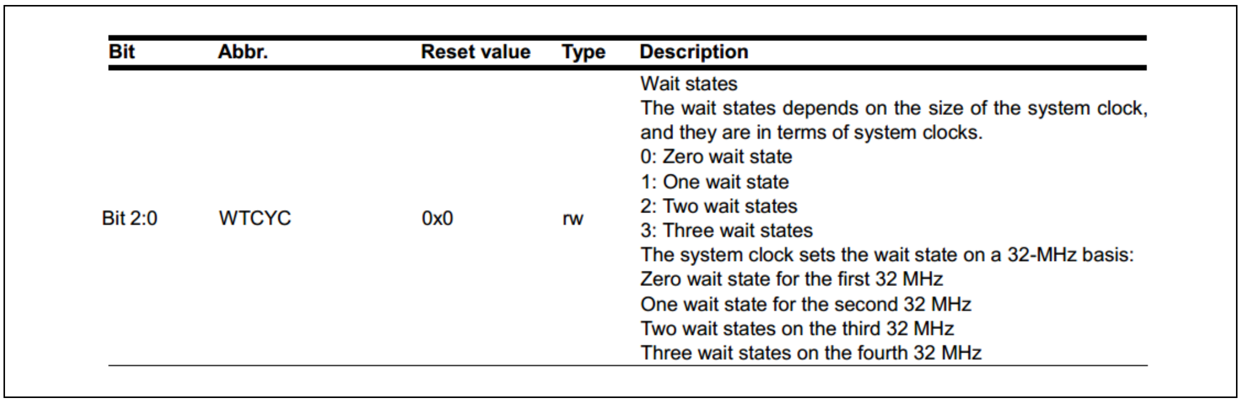
1.2 AT32F421 chip enhanced functions

1.2.1 Prefetch buffer

When the prefetch buffer is set, the CPU can execute operations faster. For example, while the CPU is reading one byte, the next byte is waiting in the prefetch buffer. The Prefetch controller decides whether to access the Flash according to the available space in the prefetch buffer. When there is at least one available block in the prefetch buffer, the prefetch controller starts a read operation.

The user needs to set wait states according to different system clock frequencies by setting bit 2:0 of the FLASH_PSR register (corresponds to the FLASH_ACR register of SXX32).

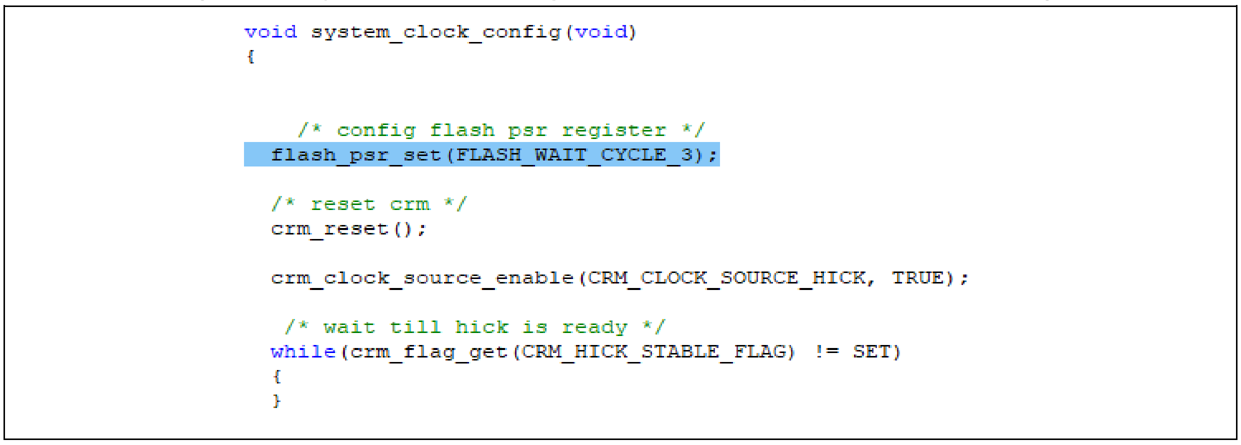
Figure 16. Wait states of Flash performance select register (FLASH_PSR)



Bit	Abbr.	Reset value	Type	Description
Bit 2:0	WTCYC	0x0	rw	Wait states The wait states depends on the size of the system clock, and they are in terms of system clocks. 0: Zero wait state 1: One wait state 2: Two wait states 3: Three wait states The system clock sets the wait state on a 32-MHz basis: Zero wait state for the first 32 MHz One wait state for the second 32 MHz Two wait states on the third 32 MHz Three wait states on the fourth 32 MHz

AT library has corresponding settings in the system clock configuration function "system_clock_config()". For other BSP versions, please find out the correct position and complete the corresponding settings.

Figure 17. System clock configuration function "system_clock_config"



```
void system_clock_config(void)
{


    /* config flash psr register */
  flash_psr_set(FLASH_WAIT_CYCLE_3);

  /* reset crm */
  crm_reset();

  crm_clock_source_enable(CRM_CLOCK_SOURCE_HICK, TRUE);

   /* wait till hick is ready */
  while(crm_flag_get(CRM_HICK_STABLE_FLAG) != SET)
  {
  }
```

1.2.2 PLL clock settings

1.2.2.1 PLL setting methods

The embedded PLL clock of AT32F421 has a maximum output frequency of 120 MHz, and it can be configured by using the CRM_CFG register (corresponds to RCC_CFGR register of SXX32) or the additional CRM_PLL register. Users can use the CRM_PLL register to configure more PLL clock frequencies, and the formula is shown below:

$$\text{PLL output clock} = \text{PLL ref. input clock} \times \frac{\text{PLL multiplication factor (PLL\_NS)}}{\text{PLL pre} - \text{division factor (PLL\_MS)} \times \text{PLL post} - \text{division factor (PLL\_FR)}}$$

When the SXX32F0xx BSP is used, the PLL setting example (HSE=8 MHz, PLL=48 MHz)

```
RCC->CFGR |= (uint32_t)(RCC_CFGR_PLLSRC_HSE | RCC_CFGR_PLLMULL6);
```

If the user wants to use SXX32F0xx program to output a clock greater than 48 MHz on AT32F421, it is necessary to configure the CRM_CFG register (corresponds to the RCC_CFGR register of SXX32).

When PLL=144 MHz, the corresponding setting is as follows:

```
RCC->CFGR |= (uint32_t)(RCC_CFGR_PLLSRC_HSE | RCC_CFGR_PLLMULL15 );
```

When the AT BSP is used, the PLL setting example (HEXT=8 MHz, PLL=48 MHz):

```
crm_pll_config(CRM_PLL_SOURCE_HEXT, CRM_PLL_MULT_6);
```

When the AT BSP is used, the PLL setting example (HEXT=8 MHz, PLL=120 MHz):

```
crm_pll_config(CRM_PLL_SOURCE_HEXT, CRM_PLL_MULT_15);
```

The user also can use the additional CRM_PLL register of AT32 MCU to configure more clock frequencies. For example, if the AT32F421 BSP is used, the PLL setting example (HEXT=8 MHz, PLL=118 MHz):

Figure 18. AT32F421 118 MHz PLL clock configuration

```
#define   CRM_PLL_NS   ((uint16_t)0x3C)   /* PLL_NS=59 */

#define   CRM_PLL_MS   ((uint16_t)0x01)   /* PLL_MS=1 */

/* config pll clock resource     PLL_FR =4*/

crm_pll_config2(CRM_PLL_SOURCE_HEXT, CRM_PLL_NS, CRM_PLL_MS, CRM_PLL_FR_4);
```

Where, the parameter CRM_PLL_SOURCE_HEXT represents that the HEXT is used as external clock source; PLL_NS=59, PLL_MS=1, CRM_PLL_FR_4 (0x02, divided by four) represents the PLL_FR value.

Please refer to *AN0116_AT32F421_CRM_Start_Guide* (ARTERY's official website→SUPPORT →AP Note→AN0116) for more details about AT32F421 clock source settings and modification and learn how to use New Clock Configuration (ARTERY's official website→PRODUCTS→Value line→AT32F4xx) to generate the desired clock code and apply it to the project.

1.2.2.2 Auto step-by-step switch

When the internal PLL of AT32F421 is set to 108 MHz and above, it is necessary to perform auto step-by-step switch.

When the SXX32F0xx BSP is used, the user needs to open system_sxx32f0xx.c and find out the current system clock configuration function (go through Section 1.2.2.1 PLL settings), and add the following codes in Italic black to the *static void SetSysClockToxxM(void)* function:

Figure 19. SXX PLL auto step-by-step switch configurations

```
/* Wait till PLL is ready */
while((RCC->CR & RCC_CR_PLLRDY) == 0)
{
}
*((unsigned int   *)0x40021054) |= (0x30);   // Enable auto step-by-step clock switch function
/* Select PLL as system clock source */
RCC->CFGR &= (uint32_t)((uint32_t)~(RCC_CFGR_SW));
RCC->CFGR |= (uint32_t)RCC_CFGR_SW_PLL;
/* Wait till PLL is used as system clock source */
while ((RCC->CFGR & (uint32_t)RCC_CFGR_SWS) != (uint32_t)0x08)
{
}
*((unsigned int   *)0x40021054) &=~ (0x30);   // Disable auto step-by-step clock switch function
```

When the AT32F421 BSP is used, the example of PLL auto step-by-step switch is shown below:

Figure 20. AT32 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 the auto step-by-step system clock switch is enabled, it must be disabled after clock switch.

1.2.3 Encryption mode

Note: The BOOT1 of AT32F421 is located in the user system data area (0x1FFF F800). When using ISP programmer, please confirm that nBOOT1=1 (default value) to allow the program to boot from boot memory (rather than SRAM).

1.2.3.1 Access protection

The access protection is commonly known as “encryption”, which is applied to the entire Flash storage area. Once the Flash access protection is enabled, the embedded Flash storage area can only be read through normal execution of the program, not through JTAG or SWD. When ICP/ISP tool is used to disable the access protection, the chip will perform erase operation on the Flash.

Note: Once enabled, the high-level access protection cannot be disabled, and the user is not allowed to re-erase or program the system data area in any way.

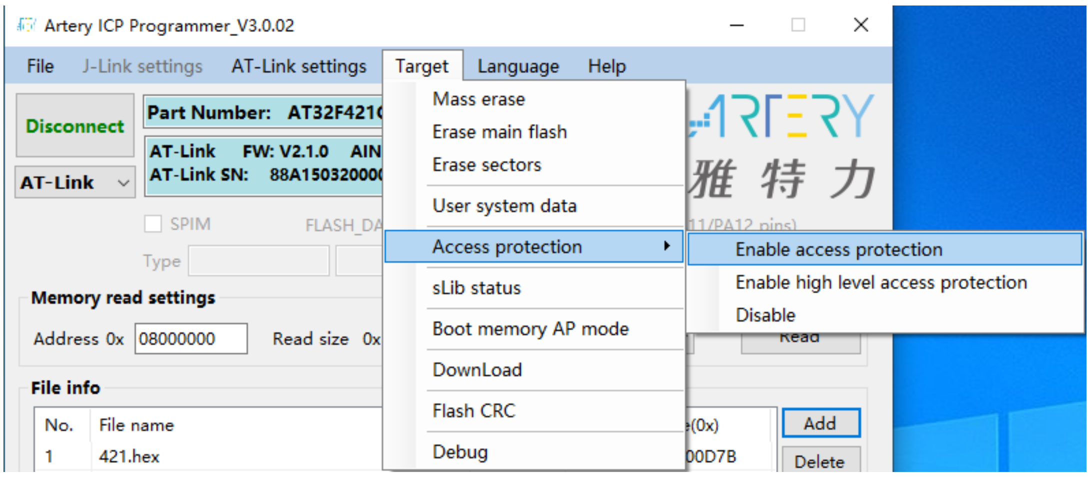
The ICP/ISP tool can be used to enable and disable IC access protection, as shown below:

- Artery ICP Programmer (BOOT0=0)

  Enable access protection: Open Artery ICP Programmer—Access protection—Enable access/high-level access protection.

  Disable access protection: Open Artery ICP Programmer—Access protection—Disable.

Figure 21. Enable/disable access protection in ICP Programmer



- Artery ISP Programmer (BOOT0=1)

  Enable access protection: Click on “Next” until enter the final interface, then select “Protection/Enable/High-level access protection” —Next—Yes, encrypted.

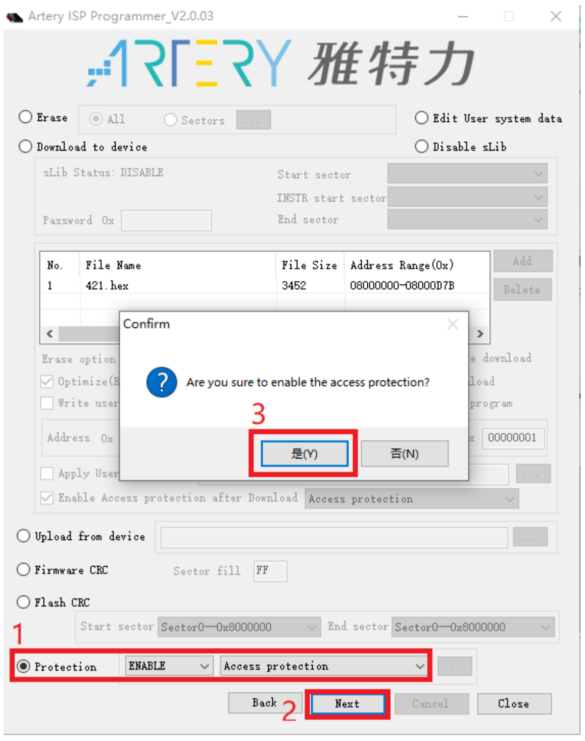
  Disable access protection: Protection/Disable/Access protection—Next—Yes, Flash decrypted.

- Artery ISP Multi-Port Programmer (BOOT0=1)

  Enable access protection: Protection/Enable/High-level access protection—Start, encrypted;

  Disable access protection: Protection/Disable/Access protection—Start, Flash decrypted.

Figure 22. Enable access protection in ISP Programmer



sLib Status: DISABLE

Password 0x

No.	File Name	File Size	Address Range(0x)
1	421.hex	3452	08000000-08000D7B

Confirm

Are you sure to enable the access protection?

是(Y) 否(N)

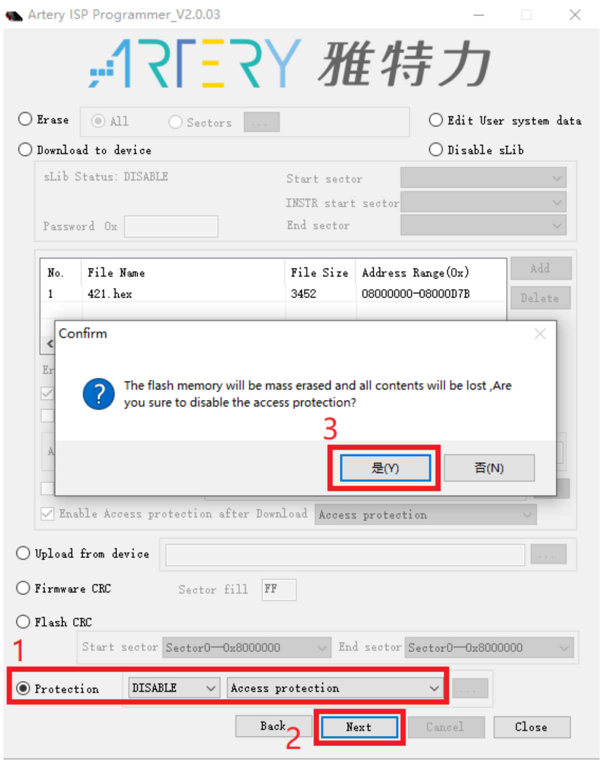
Enable Access protection after Download: Access protection

Sector fill: FF

Start sector: Sector0—0x8000000 End sector: Sector0—0x8000000

Protection: ENABLE, Access protection

Figure 23. Disable access protection in ISP Programmer



sLib Status: DISABLE

Password 0x

No.	File Name	File Size	Address Range(0x)
1	421.hex	3452	08000000-08000D7B

Confirm

The flash memory will be mass erased and all contents will be lost ,Are you sure to disable the access protection?

是(Y) 否(N)

Enable Access protection after Download: Access protection

Sector fill: FF

Start sector: Sector0—0x8000000 End sector: Sector0—0x8000000

Protection: DISABLE, Access protection

Note: Once enabled, the access protection cannot be disabled through erase operation.

1.2.3.2 Erase and program protection

The program protection is applied to the entire Flash storage area or certain pages in the Flash storage area. Once the Flash program protection is enabled, the internal Flash storage area cannot be programmed in any way.

The user can use ICP/ISP tool to enable/disable erase and program protection, as shown below:

- Artery ICP Programmer (BOOT0=0)

  Enable erase and program protection: Open Artery ICP Programmer—User system data—Tick the sectors that require erase and program protection—Apply to device.

  Disable erase and program protection: Open Artery ICP Programmer—User system data—Untick the sectors that do not require erase and program protection—Apply to device.

- Artery ISP Programmer (BOOT0=1)

  Enable erase and program protection: Protection/Enable/Erase and program protection—Next—Yes, erase and program protection enabled.

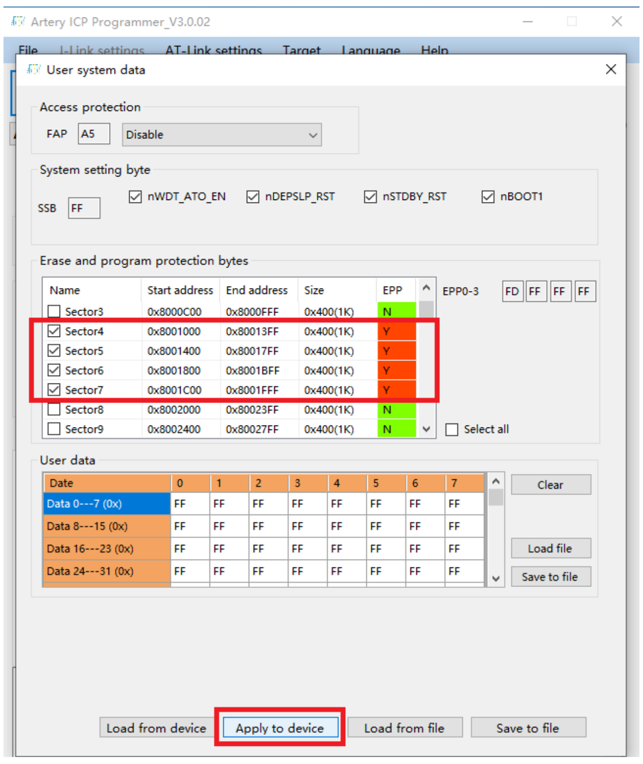
  Disable erase and program protection: Protection/Disable/Erase and program protection—Next—Yes, erase and program protection disabled.

- Artery ISP Multi-Port Programmer (BOOT0=1)

  Enable erase and program protection: Protection/Enable/Erase and program protection—Start—Yes, enabled.

  Disable erase and program protection: Protection/Disable/Erase and program protection—Start—Yes, disabled.

Figure 24. Enable erase and program protection in ICP Programmer



Access protection

FAP A5 Disable

System setting byte

SSB FF ✓ nWDT_ATO_EN ✓ nDEPSLP_RST ✓ nSTDBY_RST ✓ nBOOT1

Erase and program protection bytes

	Name	Start address	End address	Size	EPP
☐	Sector3	0x8000C00	0x8000FFF	0x400(1K)	N
✓	Sector4	0x8001000	0x80013FF	0x400(1K)	Y
✓	Sector5	0x8001400	0x80017FF	0x400(1K)	Y
✓	Sector6	0x8001800	0x8001BFF	0x400(1K)	Y
✓	Sector7	0x8001C00	0x8001FFF	0x400(1K)	Y
☐	Sector8	0x8002000	0x80023FF	0x400(1K)	N
☐	Sector9	0x8002400	0x80027FF	0x400(1K)	N

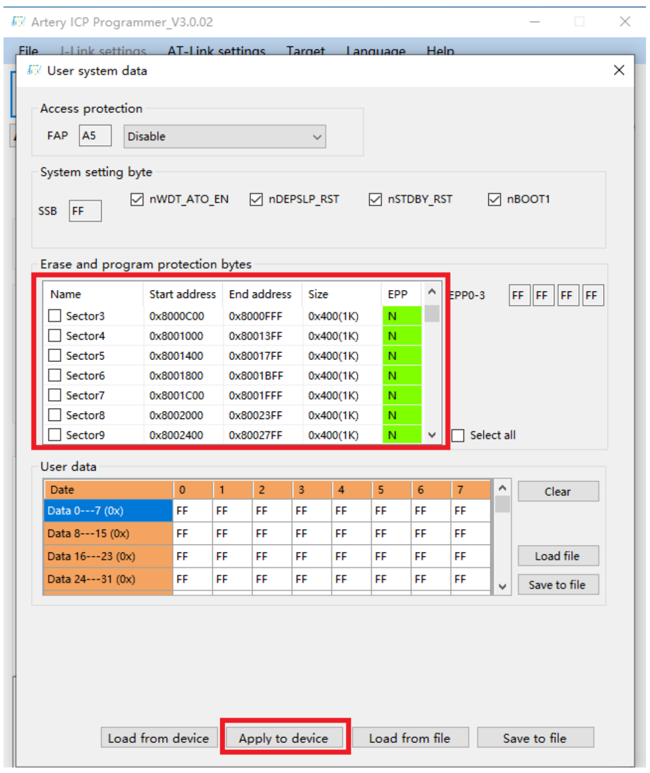
EPP0-3 FD FF FF FF

☐ Select all

User data

Date	0	1	2	3	4	5	6	7
Data 0---7 (0x)	FF	FF	FF	FF	FF	FF	FF	FF
Data 8---15 (0x)	FF	FF	FF	FF	FF	FF	FF	FF
Data 16---23 (0x)	FF	FF	FF	FF	FF	FF	FF	FF
Data 24---31 (0x)	FF	FF	FF	FF	FF	FF	FF	FF

Figure 25. Disable erase and program protection in ICP Programmer



Access protection

FAP A5 Disable

System setting byte

SSB FF ☑ nWDT_ATO_EN ☑ nDEPSLP_RST ☑ nSTDBY_RST ☑ nBOOT1

Erase and program protection bytes

Name	Start address	End address	Size	EPP
☐ Sector3	0x8000C00	0x8000FFF	0x400(1K)	N
☐ Sector4	0x8001000	0x80013FF	0x400(1K)	N
☐ Sector5	0x8001400	0x80017FF	0x400(1K)	N
☐ Sector6	0x8001800	0x8001BFF	0x400(1K)	N
☐ Sector7	0x8001C00	0x8001FFF	0x400(1K)	N
☐ Sector8	0x8002000	0x80023FF	0x400(1K)	N
☐ Sector9	0x8002400	0x80027FF	0x400(1K)	N

EPP0-3 FF FF FF FF

☐ Select all

User data

Date	0	1	2	3	4	5	6	7
Data 0---7 (0x)	FF	FF	FF	FF	FF	FF	FF	FF
Data 8---15 (0x)	FF	FF	FF	FF	FF	FF	FF	FF
Data 16---23 (0x)	FF	FF	FF	FF	FF	FF	FF	FF
Data 24---31 (0x)	FF	FF	FF	FF	FF	FF	FF	FF

Note: Once enabled, the erase and program protection cannot be disabled through erase operation.

1.2.4 Set boot memory as main Flash memory extension area

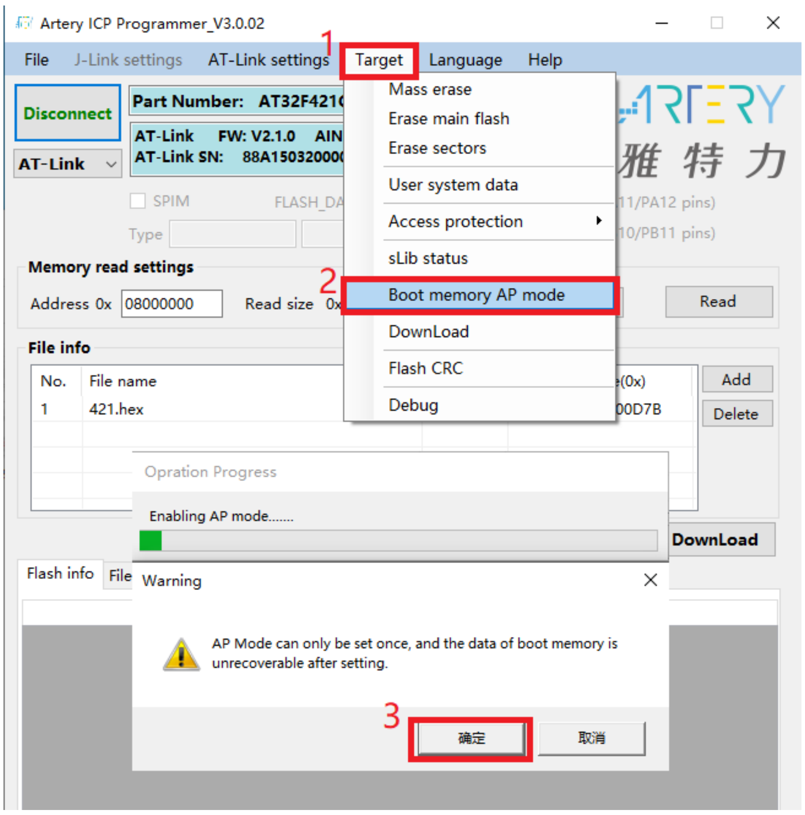
By default, boot memory stores the original boot codes in BOOT mode. For AT32F421 series MCUs, the boot memory also can be used as the main Flash memory extension area (AP mode) to store user-defined codes.

Note: The boot memory AP mode can only be set once, and the BOOT mode function of the original boot memory cannot be restored after setting.

The user can use Artery ICP Programmer to set the boot memory as the main Flash memory extension area as follows:

- Connect AT-Link/J-Link emulator to AT-START-F421 board and power on;
- Open Artery ICP programmer and select AT-Link/J-Link to connect;
- Target—Boot memory AP mode—OK.

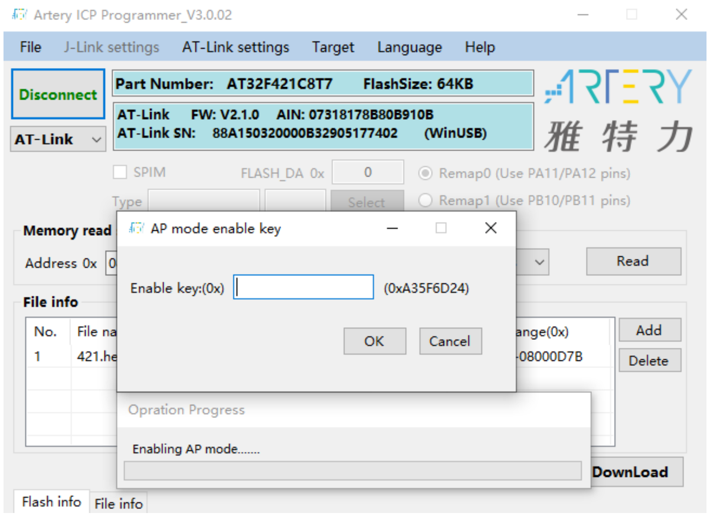
Figure 26. Set AP mode in ICP Programmer



AP Mode can only be set once, and the data of boot memory is unrecoverable after setting.

- To prevent misoperations, the user needs to manually enter the enable key 0xA35F6D24, and then the “Flash info” interface will display a success or failure message.

Figure 27. AP mode enabling in ICP Programmer



The user can use Artery ICP Programmer to set the boot memory as the main Flash memory extension area (in mass production) as follows:

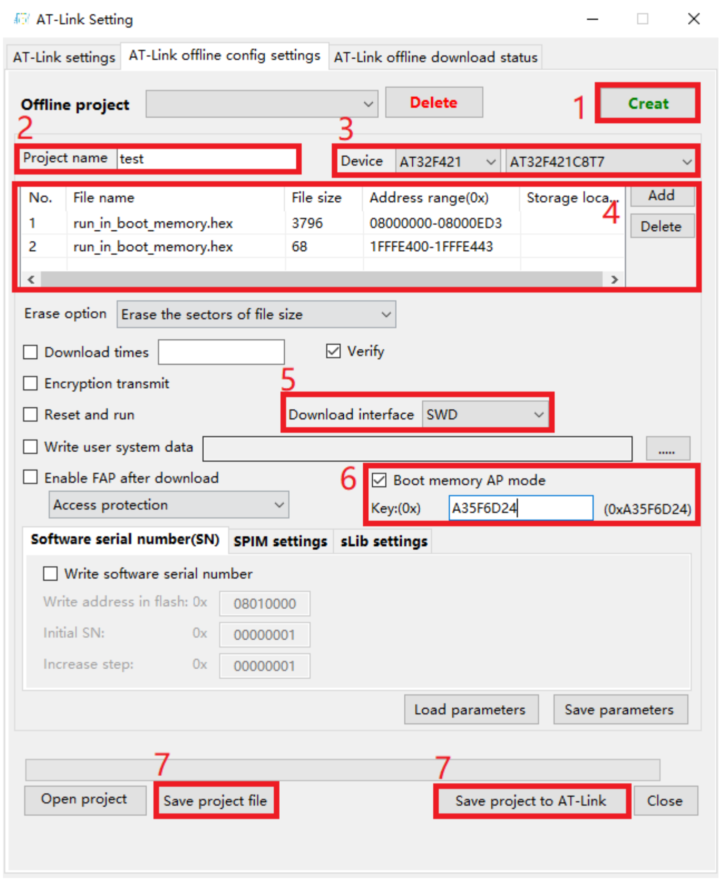
- Connect AT-Link emulator to AT-START-F421 board and power on;

Note: The on-board AT-Link EZ does not support offline programming. If necessary, please use AT-Link other than EZ version.

- Open Artery ICP programmer and select AT-Link to connect;
- AT-Link Setting—AT-Link offline config settings;
- Generate an offline project as follows:
  1. Click on “Create”;
  2. Enter project name;
  3. Select MCU part number;
  4. Add .hex file
  5. Select SWD as the download interface;
  6. Tick boot memory AP mode and enter the key;
  7. Save project to AT-Link or save project file.

Complete other relevant settings as necessary.

Figure 28. Offline config settings in ICP Programmer



AT-Link Setting

AT-Link settings | AT-Link offline config settings | AT-Link offline download status

Offline project | Delete | 1 Creat

2 Project name: test

3 Device: AT32F421 | AT32F421C8T7

4

No.	File name	File size	Address range(0x)	Storage loca...
1	run_in_boot_memory.hex	3796	08000000-08000ED3	
2	run_in_boot_memory.hex	68	1FFFE400-1FFFE443	

Add | Delete

Erase option: Erase the sectors of file size

☐ Download times ☑ Verify

☐ Encryption transmit

☐ Reset and run 5 Download interface: SWD

☐ Write user system data .....

☐ Enable FAP after download 6 ☑ Boot memory AP mode

Access protection Key:(0x) A35F6D24 (0xA35F6D24)

Software serial number(SN) | SPIM settings | sLib settings

☐ Write software serial number

Write address in flash: 0x 08010000

Initial SN: 0x 00000001

Increase step: 0x 00000001

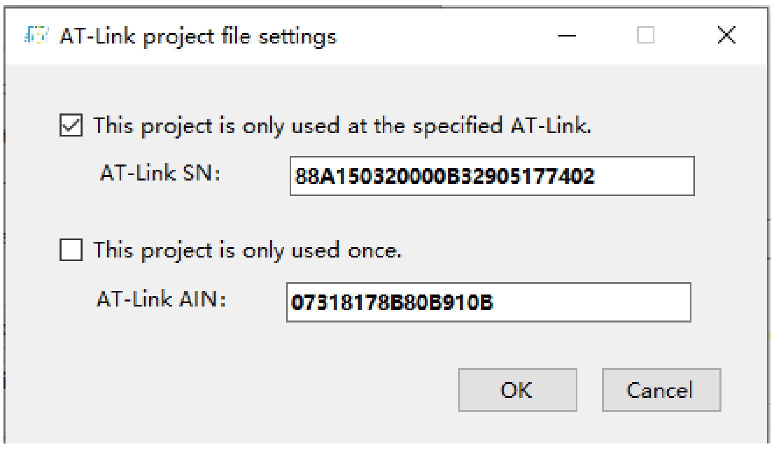
Load parameters | Save parameters

Open project | 7 Save project file | 7 Save project to AT-Link | Close

- If you select “Save project file”, the project will be saved as an .atcp file for easy loading to other versions of AT-Link.

  The following window (as shown in Figure 29) will pop up in operation. If you tick “This project is only used at the specific AT-Link”, this project file will be bound to a specific AT-Link and can only be used on this AT-Link, and you need to set the AT-Link serial number. If you tick “This project is only used once”, this project file can only be used once on the same AT-Link.

Figure 29. AT-Link project file settings



AT-Link project file settings

☑ This project is only used at the specified AT-Link.

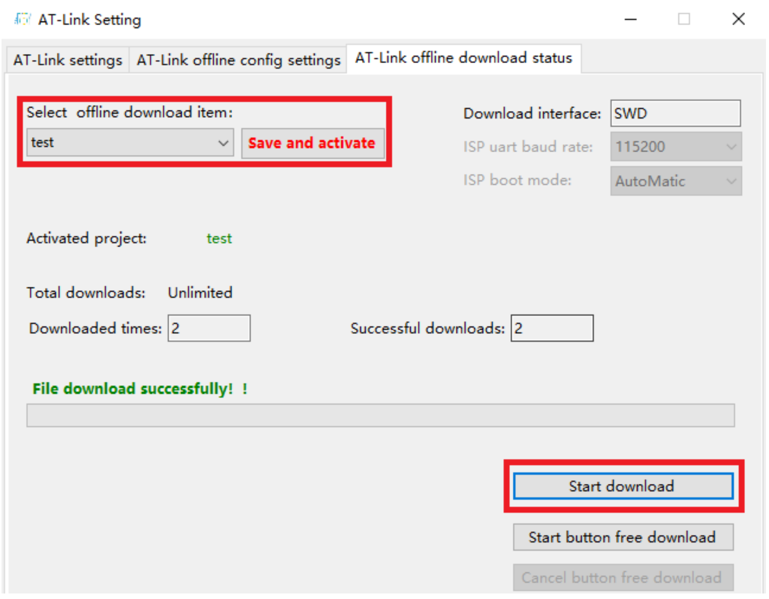
AT-Link SN: 88A150320000B32905177402

☐ This project is only used once.

AT-Link AIN: 07318178B80B910B

- If the project is saved to AT-Link successfully, in the offline download status interface (as shown in Figure 30), select offline download item—Save and activate, to start download.

Figure 30. AT-Link offline download status



AT-Link Setting

AT-Link settings | AT-Link offline config settings | AT-Link offline download status

Select offline download item: test — Save and activate

Download interface: SWD

ISP uart baud rate: 115200

ISP boot mode: AutoMatic

Activated project: test

Total downloads: Unlimited

Downloaded times: 2

Successful downloads: 2

File download successfully! !

- Refer to *AN0066_config_boot_memory_as_extension_of_main_memory (AP_mode)* (ARTERY's official website→SUPPORT→AP Note→AN0066) for more information about the memory extension.
- Refer to BSP for the demo of running user application in boot memory. Path: ARTERY's official website→PRODUCTS→Value line→AT32F4x; download and unzip BSP, and then get the “AT32F421_Firmware_Library_V2.x.x\utilities\at32f421_boot_memory_ap_demo”.

1.2.5 Recognize AT32 MCU in program

■ Read Cortex-M CPU ID to identify M0, M3 and M4 core

Figure 31. 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 PID and UID

Figure 32. Read PID and UID

```
/* Get the base address of AT32 MCU PID/UID */
#define DEVICE_ID_ADDR1 0x1FFFF7F3            // Define Artery MCU part number, UID base address
#define DEVICE_ID_ADDR2 0xE0042000            // Define MCU device number, PID base address

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

/* AT32F421 MCU type table */
const uint64_t AT32_MCU_ID_TABLE[] =
{
      0x0000000950010016,   // AT32F421PF4P7      16KB     TSSOP20
      0x0000000950020115,   // AT32F421PF8P7      64KB     TSSOP20
      …
};

  /* Get PID/UID */
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 PID/UID */
  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);

/* Judge 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: The AT32F4xx MCU contains multiple ID codes. Combine the obtained ID information to a 64-bit data to help identify the specific MCU model. For more information, please refer to the

Reference Manual (DEBUG section) of each series and *AN0016_Recognize_AT32_MCU* (ARTERY's official website→SUPPORT→AP Note→AN0016).

2 FAQs in downloading/compiling

2.1 Hard Fault Handler

- The data being accessed is out of range.

  Find out the access violation point and modify the data to the normal range.
- The SRAM being used exceeds the specified MCU SRAM.
- The system clock configuration is out of specification.

2.2 J-Link cannot find IC

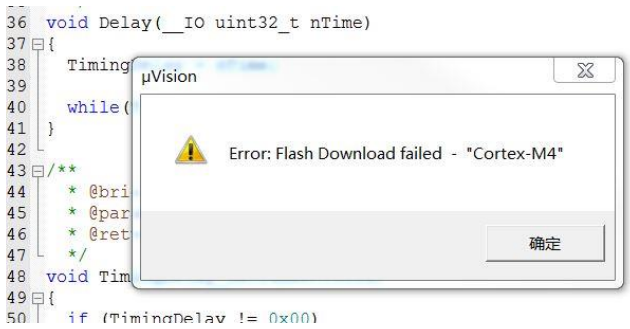
- Please refer to *FAQ0008_ J-Link cannot find IC* (ARTERY's official website→ SUPPORT→FAQ→FAQ0008).
- Please refer to *FAQ0132_Add Artery MCU to J-Link* (ARTERY's official website→ SUPPORT→FAQ→FAQ0132).

2.3 Errors in program downloading

2.3.1 Flash Download failed–"Cortex-M4"

The following pop-up window appears in KEIL emulation or downloading:

Figure 33. Flash Download failed–"Cortex- M4" pops up in downloading



```
void Delay(__IO uint32_t nTime)
{
  Timing

  while (
}

/**
  * @bri
  * @par
  * @ret
  */
void Tim
{
  if (TimingDelay != 0x00)
```

µVision

Error: Flash Download failed - "Cortex-M4"

This pop-up window may appear in one of the following conditions:

- The access protection is enabled. Please disable MCU access protection and then download
- The selected Flash file algorithm is incorrect or no Flash file algorithm is loaded. Please add the proper Flash file algorithm in "Flash Download".
- BOOT0 is incorrect. Set BOOT0=0 to allow MCU to boot from main Flash memory.
- The J-Link version is outdated. Please use 6.20C and above.
- JTAG/SWD PIN is disabled in the program. Please refer to "2.3.5 Resume download".

2.3.2 No Debug Unit Device found

- The download port is occupied. For example, ICP is connecting to the target device.
- JTAG/SWD connection error or not connected.

2.3.3 RDDI-DAP Error

- Compiler optimization level is too high. For example, the default optimization level of Keil AC6 is –Oz, which should be modified to -O0/-O1.
- JTAG/SWD PIN is disabled in the program. Please refer to "2.3.5 Resume download".

2.3.4 ISP serial port gets stuck in downloading

When the ISP serial port is used for downloading, it may be stuck occasionally, and the PC cannot release serial port in this case.

The following operations are recommended:

- Check whether the power supply is stable;
- Replace with better USB-to-serial port tools, i.e., CH340 chip.

2.3.5 Resume download

When using AT32F421, the user may not be able to download the program again in the following conditions:

- After disabling JTAG/SWD PIN in the program, the program cannot be downloaded and JTAG/SWD device is not found.
- After entering Standby mode the program cannot be downloaded and JTAG/SWD device is not found.

Solutions:

- ➢ Solution 1: Switch boot mode.
  Switch boot mode to Boot0=1, and then press "Reset" button to resume download (note: switch back to Boot0=0). Similarly, ISP download can be resumed.
- ➢ Solution 2: ICP tool + AT-Link.
  AT-Link is specially designed for AT32 MCUs. The ICP can be used together with AT-Link to resume download.

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.

The AT32F421 series is designed with a security library (sLib) to protect important IP-Codes against being changed or read by the end user’s program.

3.2 Principles of application

- 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 terminal customers.
- Security library includes the read-only area (SLIB_READ_ONLY) and instruction security library (SLIB_INSTRUCTION), and it can be completely or partially used as read-only area or instruction security library.
- Data in the read-only area (SLIB_READ_ONLY) can be read through I-Code and D-Code bus and cannot be programmed.
- Program code in the instruction security library (SLIB_INSTRUCTION) can only be fetched by MCU through I-Code bus (can only be executed) and cannot be read through D-Code bus (including ISP/ICP debug mode and programs that boot from internal RAM). When accessing the SLIB_INSTRUCTION in the manner of reading data, values are all read 0xFF.
- The program code 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 erasing the security library, a warning message will be issued by EPPERR=1 in the FLASH_STS register.
- The program code and data in security library are not erased when the end users perform a mass erase on the main Flash memory.
- Users can write the previously defined password in the SLIB_PWD_CLR register to disable security library protection. When the security library protection is disabled, the chip will perform a mass erase on the main Flash memory (including the contents of security library). Therefore, even if the code defined by the software solution provider is leaked, the program code will not be leaked.

3.3 Security library application

For details, please refer to *AN0071_AT32F421_Security_Library_Application_Note* (ARTERY's official website→SUPPORT→AP Note→AN0071).

4 Revision history

Table 1. Document revision history

Date	Version	Revision note
2021.12.29	2.0.0	Initial release
2022.05.09	2.0.1	Added description of BOOT1 and a prompt of high-level access protection cannot be disabled; optimized contents.
2022.07.15	2.0.2	Optimized contents.
2022.10.10	2.0.3	Updated 3rd party programming tools and added description of development environment and file path.
2022.10.21	2.0.4	Optimized description of UID and PID.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY’s products and services; ARTERY assumes no liability for purchasers’ selection or use of the products and the relevant services.

No license, express or implied, to any intellectual property right is granted by ARTERY herein regardless of the existence of any previous representation in any forms. If any part of this document involves third party’s products or services, it does NOT imply that ARTERY authorizes the use of the third party’s products or services, or permits any of the intellectual property, or guarantees any uses of the third party’s products or services or intellectual property in any way.

Except as provided in ARTERY’s terms and conditions of sale for such products, ARTERY disclaims any express or implied warranty, relating to use and/or sale of the products, including but not restricted to liability or warranties relating to merchantability, fitness for a particular purpose (based on the corresponding legal situation in any unjudicial districts), or infringement of any patent, copyright, or other intellectual property right.

ARTERY’s products are not designed for the following purposes, and thus not intended for the following uses: (A) Applications that have specific requirements on safety, for example: life-support applications, active implant devices, or systems that have specific requirements on product function safety; (B) Aviation applications; (C) Auto-motive application or environment; (D) Aerospace applications or environment, and/or (E) weapons. Since ARTERY products are not intended for the above-mentioned purposes, if purchasers apply ARTERY products to these purposes, purchasers are solely responsible for any consequences or risks caused, even if any written notice is sent to ARTERY by purchasers; in addition, purchasers are solely responsible for the compliance with all statutory and regulatory requirements regarding these uses.

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.

© 2022 ARTERY Technology – All Rights Reserved