AN0129
Application Note

Getting Started with AT32F425

Introduction

This document is aimed at helping users with project development based on AT32F425xx MCUs.

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	AT32F425xx

Contents

1 Development resources 5

1.1 Set up AT32 development environment 5

1.1.1 Debug tools and evaluation board 5

1.1.2 Programming tools and software resources 5

1.1.3 AT32 development environment 6

1.1.4 How to quickly replace AT32F415 with AT32F425 12

1.2 AT32F425 functionality configuration 12

1.2.1 Instruction prefetch buffer 12

1.2.2 PLL clock configuration 13

1.2.3 Encryption 14

1.2.4 Set system memory as main memory extension 17

1.2.5 How to distinguish AT32 MCU from other MCUs 21

2 FAQs about download and compiling 22

2.1 Program enters Hard Fault Handler 22

2.2 JLink unable to recognize IC in Keil project 22

2.3 Errors during download 22

2.3.1 Flash Download failed–“Cortex-M4” 22

2.3.2 No Debug Unit Device found 23

2.3.3 RDDI-DAP Error 23

2.3.4 ISP serial interface gets stuck during download 23

2.3.5 How to resume program download 23

3 Security Library (sLib) 24

3.1 Introduction 24

3.2 Application principles 24

3.3 How to use sLib 24

4 Revision history 25

List of tables

Table 1. Document revision history 25

List of figures

Figure 1. AT_START_F425 and AT-Link-EZ 5
Figure 2. AT-START-F425 evaluation board package from ARTERY official website 5
Figure 3. ICP/ISP/AT-Link-Family package on ARTERY official website 6
Figure 4. BSP resources on ARTERY official website 6
Figure 5. Keil_v5 templates template 7
Figure 6. Pack resources on ARTERY official website 8
Figure 7. Install ArteryTek.AT32F425_DFP 8
Figure 8. Install Keil4_AT32MCU_AddOn 8
Figure 9. Click “Pack Installer” in Keil 9
Figure 10. Install IAR_AT32MCU_AddOn 9
Figure 11. Keil Debug option 10
Figure 12. Settings option in Keil Debug 10
Figure 13. Keil Utilities option 11
Figure 14. IAR Debug option 11
Figure 15. IAR CMSIS-DAP option 12
Figure 16. Wait state bit in FLASH_PSR register 13
Figure 17. system_clock_config function 13
Figure 18. AT32F425 94MHz output clock 13
Figure 19. Use ICP tool to enable/disable access protection 14
Figure 20. Use ISP to enable access protection 15
Figure 21. Use ISP to disable access protection 15
Figure 22. Use ICP tool to enable erase/program protection 16
Figure 23. Use ICP tool to disable erase/program protection 17
Figure 24. Use ICP tool to set boot memory AP mode 18
Figure 25. Boot memory AP mode operating progress 18
Figure 26. Use ICP tool to set offline boot mode AP mode 19
Figure 27. Use ICP tool to set offline project programming 20
Figure 28. ICP tool to monitor offline download status 20
Figure 29. Read Cortex ID 21
Figure 30. Read PID and UID 21
Figure 31. Error: Flash Download failed–“Cortex- M4” 22

1 Development resources

Resources download link:

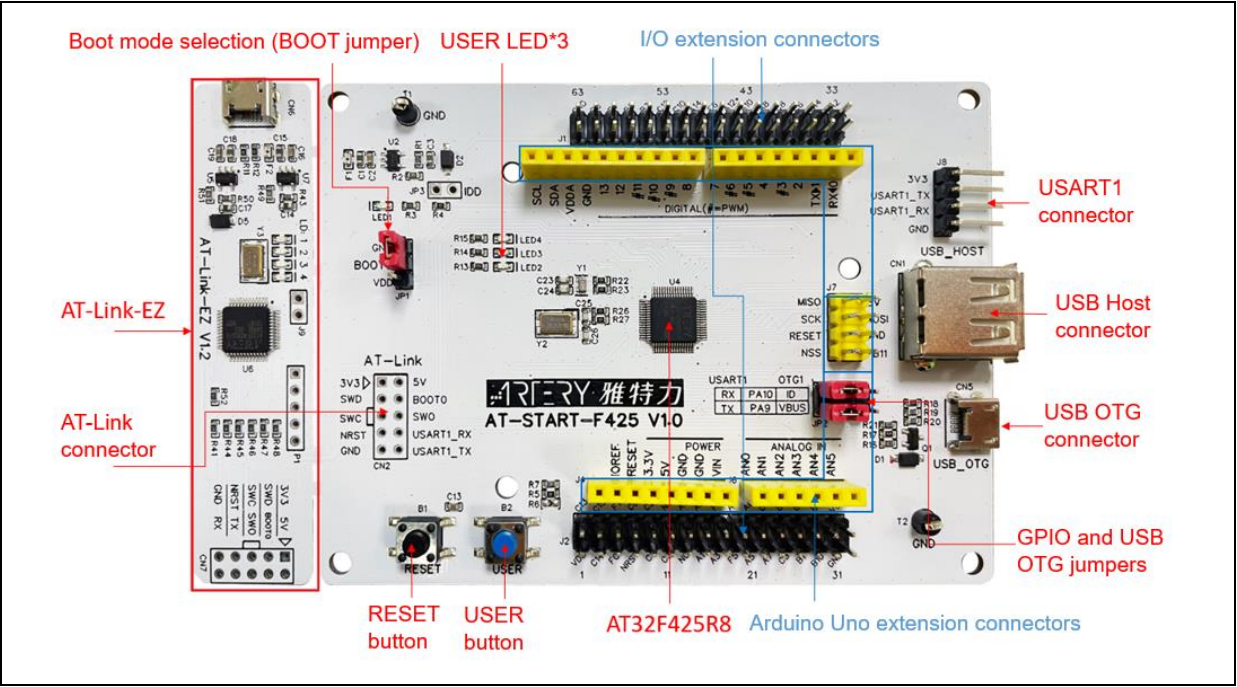
- Visit Artery website: https://www.arterychip.com

1.1 Set up AT32 development environment

1.1.1 Debug tools and evaluation board

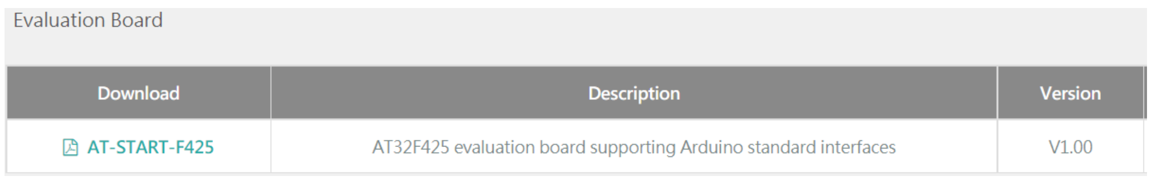
The AT32F425 evaluation board comes with AT-Link-EZ for the debug purpose, as shown in the left red box of the figure below. It can also be disassembled from the evaluation board and used separately with other circuit boards. This debug tool can be used for several purposes such IDE online debugging, online programming and USB-to-serial interface.

Figure 1. AT_START_F425 and AT-Link-EZ



Note: For details on AT-START-AT32F425 evaluation board, refer to the "UM_AT_START_F425_Vx.x", which is available from [ARTERY official website](https://www.arterychip.com)→Product→Value line→AT32F425 series→Resources→Evaluation Board (download and unzip, and then go to "\AT_START_F425_Vx.x\03_Documents").

Figure 2. AT-START-F425 evaluation board package from ARTERY official website



Evaluation Board

Download	Description	Version
AT-START-F425	AT32F425 evaluation board supporting Arduino standard interfaces	V1.00

1.1.2 Programming tools and software resources

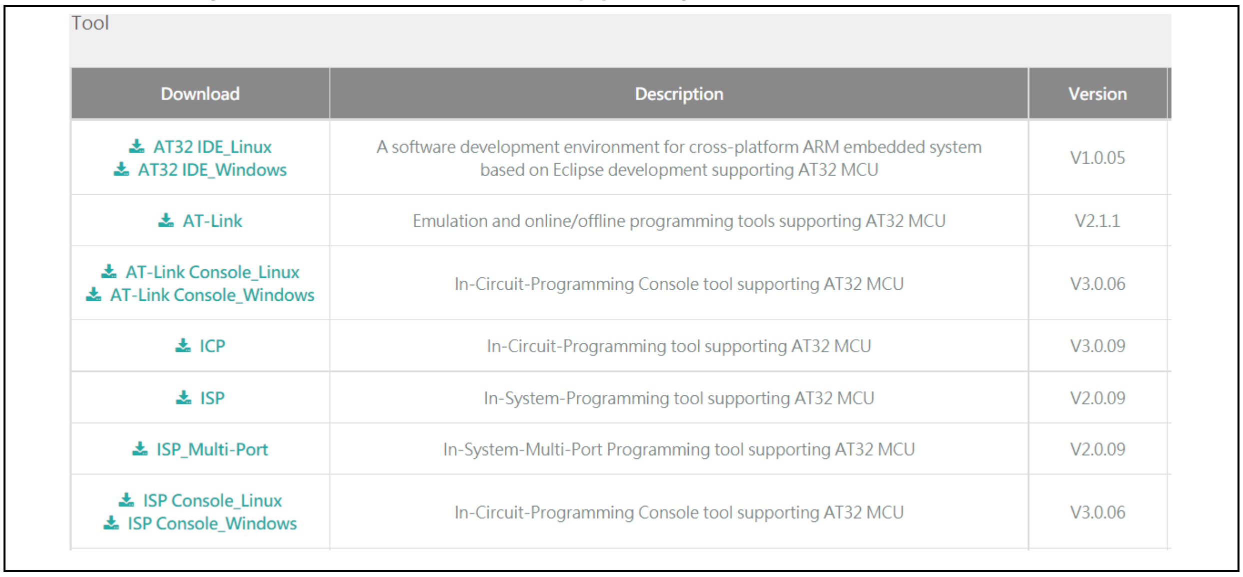
- AT programming tools and software: AT-Link / AT-Link+ /AT-Link-Pro / AT-Link-ISO /AT-Link-EZ, and ICP/ISP

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

Note: These programming tools are available from ARTERY official website → Support→Hardware Development Tool or 3rd Party Writer.

- **ICP user guide:** Refer to the *UM_ICP_Programmer*, which is available from ARTERY official website→Products→Value line→AT32F4xx→Tool→ICP (download and unzip, and then go to Artery_ICP_Programmer_Vx.x.xx\Document\UM_ICP_Programmer)
- **ISP user guide:** Refer to the *UM_ISP_Programmer*, which is available from ARTERY official website→Products→Value line→AT32F4xx→Tool→ISP (download and unzip, and then go to Artery_ISP_Programmer_Vx.x.xx\Document\UM_ISP_Programmer)
- **AT-Link user guide:** Refer to the *UM0004_AT-Link_User_Manual*, which is available from ARTERY official website → Products→Value line→AT32F4xx→Tool→AT-Link-Family (download and unzip, and then go to AT_Link_CH_ Vx.x.x\05_Documents\UM0004_AT-Link_User_Manual_EN_Vx.x.x)

Figure 3. ICP/ISP/AT-Link-Family package on ARTERY official website



Tool

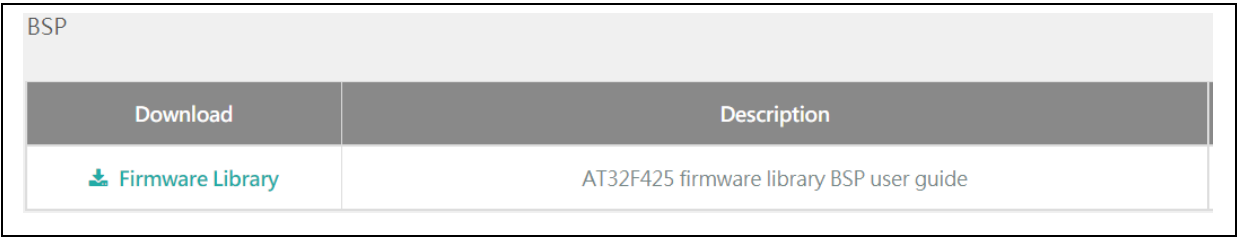
Download	Description	Version
AT32 IDE_Linux AT32 IDE_Windows	A software development environment for cross-platform ARM embedded system based on Eclipse development supporting AT32 MCU	V1.0.05
AT-Link	Emulation and online/offline programming tools supporting AT32 MCU	V2.1.1
AT-Link Console_Linux AT-Link Console_Windows	In-Circuit-Programming Console tool supporting AT32 MCU	V3.0.06
ICP	In-Circuit-Programming tool supporting AT32 MCU	V3.0.09
ISP	In-System-Programming tool supporting AT32 MCU	V2.0.09
ISP_Multi-Port	In-System-Multi-Port Programming tool supporting AT32 MCU	V2.0.09
ISP Console_Linux ISP Console_Windows	In-Circuit-Programming Console tool supporting AT32 MCU	V3.0.06

1.1.3 AT32 development environment

1.1.3.1 Template project

The frequently-used IDE template projects are included in ARTERY's firmware BSP. You can get these resources by visiting ARTERY official website→Products→Value line→AT32F425 series→BSP.

Figure 4. BSP resources on ARTERY official website



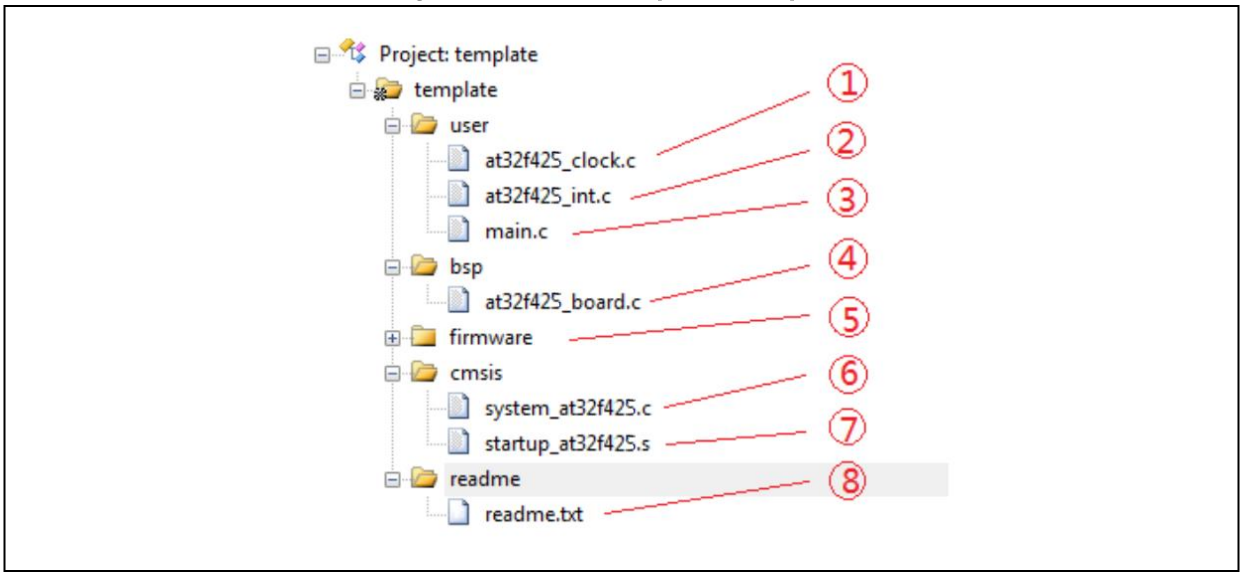
BSP

Download	Description
Firmware Library	AT32F425 firmware library BSP user guide

BSP offers such template projects as *Keil_v5/Keil_v4/IAR_6.10/IAR_7.4/IAR_8.2/eclipse_gcc /at32_ide*, which are stored at AT32F425_Firmware_Library_V2.x.x\project\at_start_f4xx \templates.

Just simply open the desired folder and IDE project. Figure 5 below shows an example of Keil_v5 project.

Figure 5. Keil_v5 templates template



This template mainly contains the following items:

① **at32f425_clock.c**: clock configuration file, which defines the default clock frequency and clock path

② **at32f425_int.c**: interrupt file, which contains the default process of handling some core interrupts

③ **main.c**: main code file

④ **at32f425_board.c**: board configuration file, which defines common hardware such as buttons and LEDs on AT-START evaluation board

⑤ **firmware**: it contains "*at32f425_xx.c*" that is used as a driver file for peripherals

⑥ **system_at32f425.c**: system initialization file

⑦ **startup_at32f425.s**: startup file

⑧ **readme.txt**: a read-me txt file that describes some application functions, configuration method and application notes relating to a template project.

In addition to "Templates", a large number of examples are included in BSP for users' reference. These example codes are stored at:

AT32F425_Firmware_Library_V2.x.x\project\at_start_f4xx\examples.

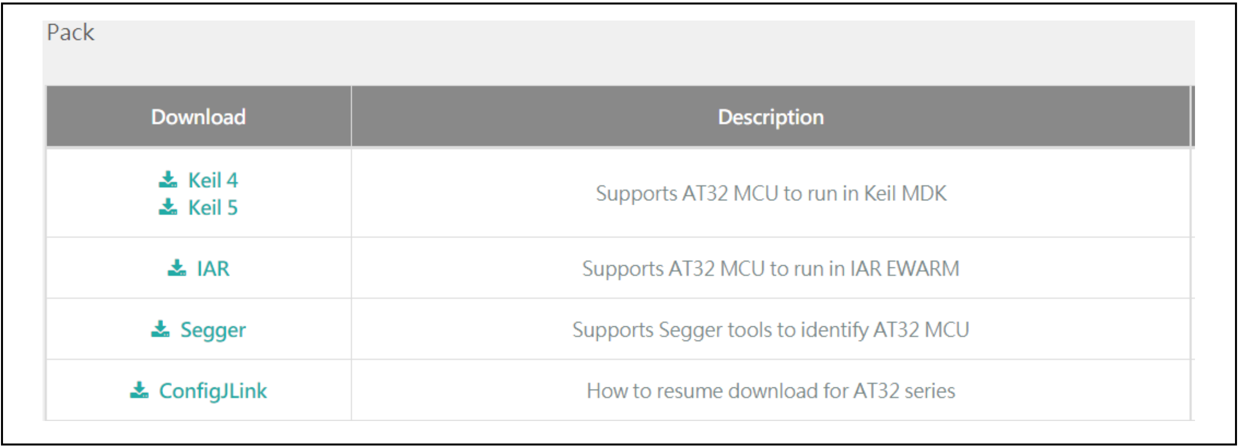
Note: For details on BSP, refer to Section 4 of the document "AT32F425_firmware_BSP&Pack_user_guide", which is available from ARTERY official website→Products→Value line→AT32F425 series→BSP (download and unzip, and then go to "\AT32F425_Firmware_Library_Vx.x.x\document").

1.1.3.2 Pack installation

The installation of Pack is required to add AT32 MCU part number in Keil/IAR.

The Pack is available from ARTERY official website→Products→Value line→AT32F425 series.

Figure 6. Pack resources on ARTERY official website

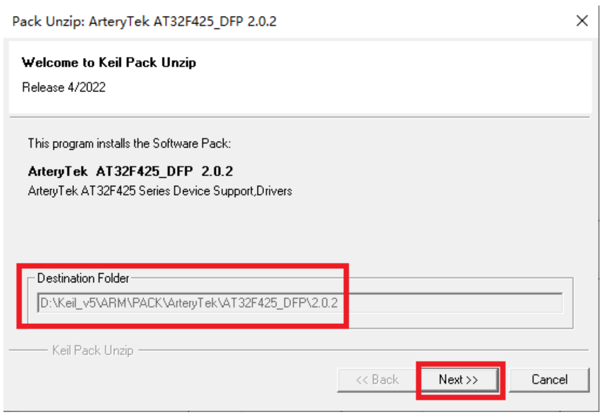


Pack

Download	Description
Keil 4 Keil 5	Supports AT32 MCU to run in Keil MDK
IAR	Supports AT32 MCU to run in IAR EWARM
Segger	Supports Segger tools to identify AT32 MCU
ConfigJLink	How to resume download for AT32 series

For the Keil compiling system, Keil 4.74, Keil 5.23 and above versions are recommended.
For Keil_v5, users need to unzip the "*Keil5_AT32MCU_AddOn*" and then install the "*ArteryTek.AT32F425_DFP*".
For Keil_v4, users directly install the "*Keil4_AT32MCU_AddOn*".
By default, the installation path of Keil can be automatically recognized when installing. In case of recognition failure, users need to manually select the Keil installation path.

Figure 7. Install ArteryTek.AT32F425_DFP



Pack Unzip: ArteryTek AT32F425_DFP 2.0.2

Welcome to Keil Pack Unzip

Release 4/2022

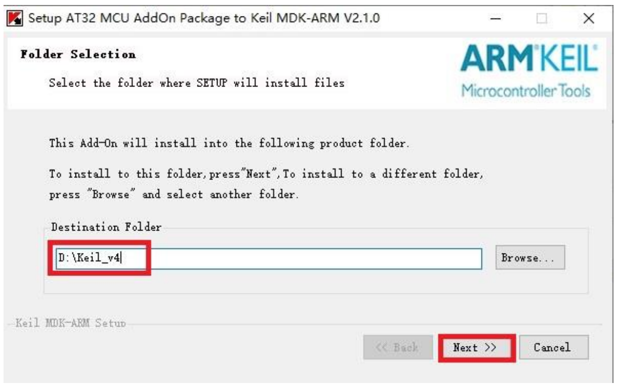
This program installs the Software Pack:

ArteryTek AT32F425_DFP 2.0.2

ArteryTek AT32F425 Series Device Support,Drivers

Destination Folder: D:\Keil_v5\ARM\PACK\ArteryTek\AT32F425_DFP\2.0.2

Figure 8. Install Keil4_AT32MCU_AddOn



Setup AT32 MCU AddOn Package to Keil MDK-ARM V2.1.0

Folder Selection

Select the folder where SETUP will install files

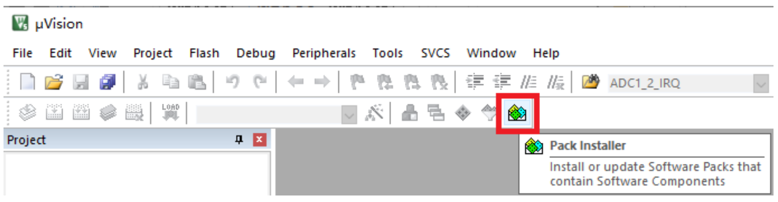
This Add-On will install into the following product folder.

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

Destination Folder: D:\Keil_v4

Users can also open Keil→click "Pack Installer"→click "File"→choose "Import" to import the corresponding pack downloaded from ARTERY official website.

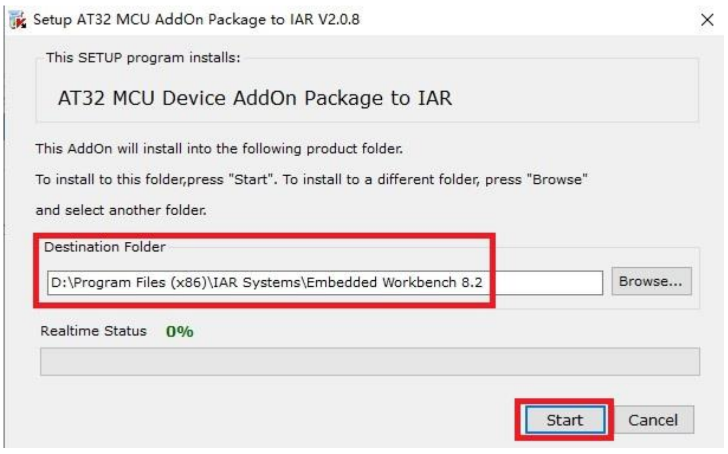
Figure 9. Click "Pack Installer" in Keil



For the IAR compiling system, IAR7.0, IAR6.1 and above versions are recommended.

By default, when installing the "*IAR_AT32MCU_AddOn*", the installation path of IAR can be automatically recognized. In case of recognition failure, users need to manually select the IAR installation path.

Figure 10. Install IAR_AT32MCU_AddOn


Setup AT32 MCU AddOn Package to IAR V2.0.8

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

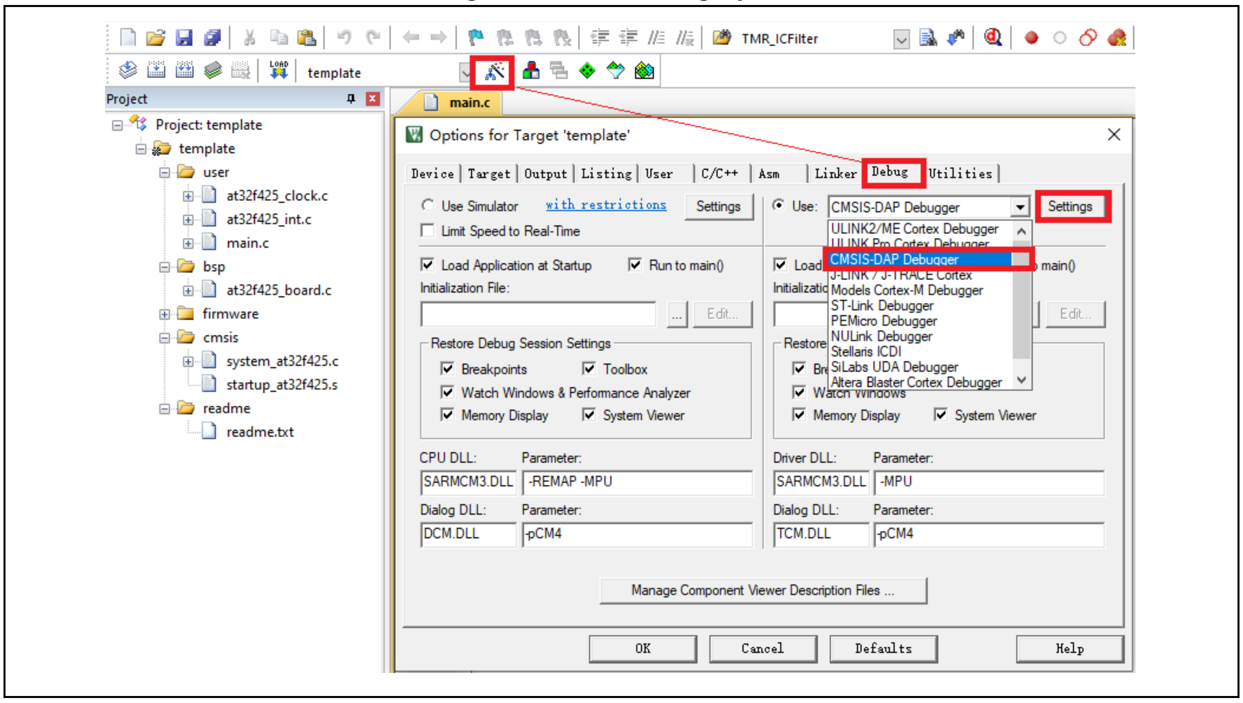
Realtime Status 0%

Note: For details on Pack, refer to Section 2 of the document "AT32F425_firmware_BSP&Pack_user_guide", which is available from ARTERY official website →Products→Value line→AT32F425 series→BSP (download and unzip, and then go to "\AT32F425_Firmware_Library_Vx.x.x\document").

1.1.3.3 Debug and download with AT-Link tool

If AT-Link is to be used in Keil environment, click "Debug", and then choose "CMSIS-DAP Debugger".

Figure 11. Keil Debug option



Options for Target 'template'

Device | Target | Output | Listing | User | C/C++ | Asm | Linker | Debug | Utilities

( ) 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
[✓] Memory Display [✓] System Viewer
Driver DLL: SARMCM3.DLL Parameter: -MPU
Dialog DLL: TCM.DLL Parameter: -pCM4

Manage Component Viewer Description Files ...
OK | Cancel | Defaults | Help

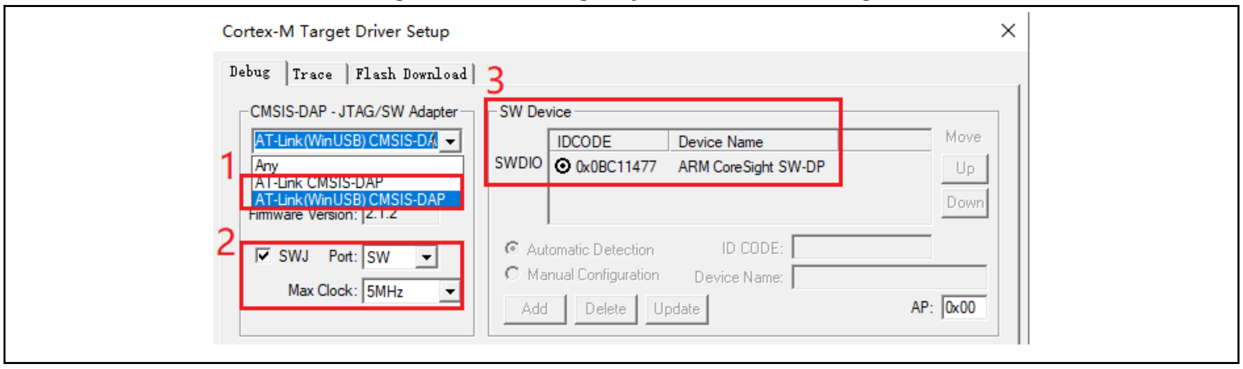
Next, click "Settings" to enter "Cortex-M Target Driver Setup" window, as shown in Figure 12 below.

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

Note: For more information on WinUSB, refer to "FAQ0136_How_to_use_AT-LINK_WinUSB_EN_V2.0.0", which is available from [*ARTERY official website*]*→Support→FAQ→FAQ0136.*

2. In "Port" option, choose "SW", and then tick "SWJ" option;
3. Confirm that the "ARM SW-DP" is recognized.

Figure 12. Settings option in Keil Debug



Cortex-M Target Driver Setup

Debug | Trace | Flash Download

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

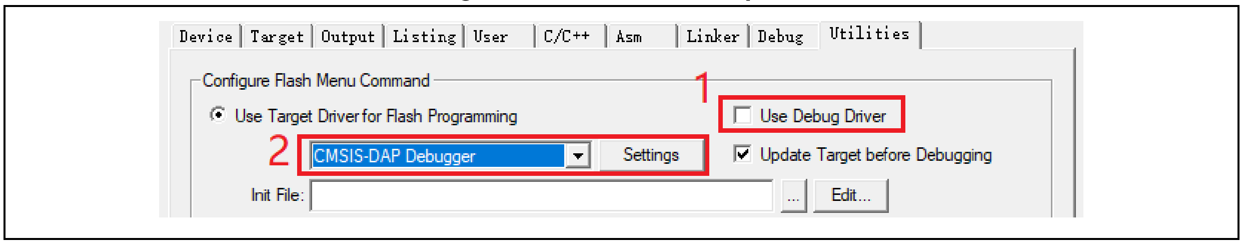
SW Device

	IDCODE	Device Name
SWDIO	0x0BC11477	ARM CoreSight SW-DP

(•) Automatic Detection ID CODE:
( ) Manual Configuration Device Name:
Add | Delete | Update AP: 0x00

Then, click "Utilities"→untick "Use Debug Driver" (the red box marked 1)→ select "CMSIS-DAP Debugger" (the red box marked 2) →finally tick "Use Debug Driver" (note: users need to untick this box first and then tick it).

Figure 13. Keil Utilities option



Configure Flash Menu Command

(●) Use Target Driver for Flash Programming

☐ Use Debug Driver

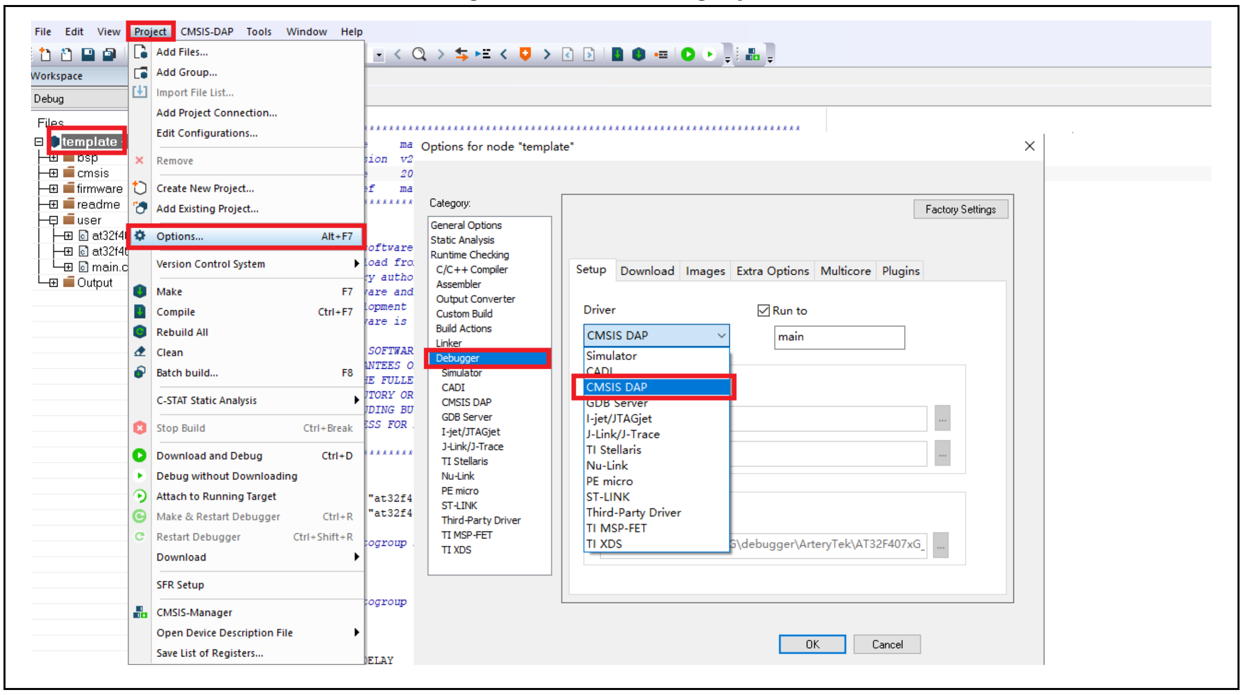
CMSIS-DAP Debugger

☑ Update Target before Debugging

Init File:

If AT-Link is to be used in IAR environment, click “Project”→choose “Options” →go to “Debugger” and choose “CMSIS-DAP”→tick “SWD”.

Figure 14. IAR Debug option

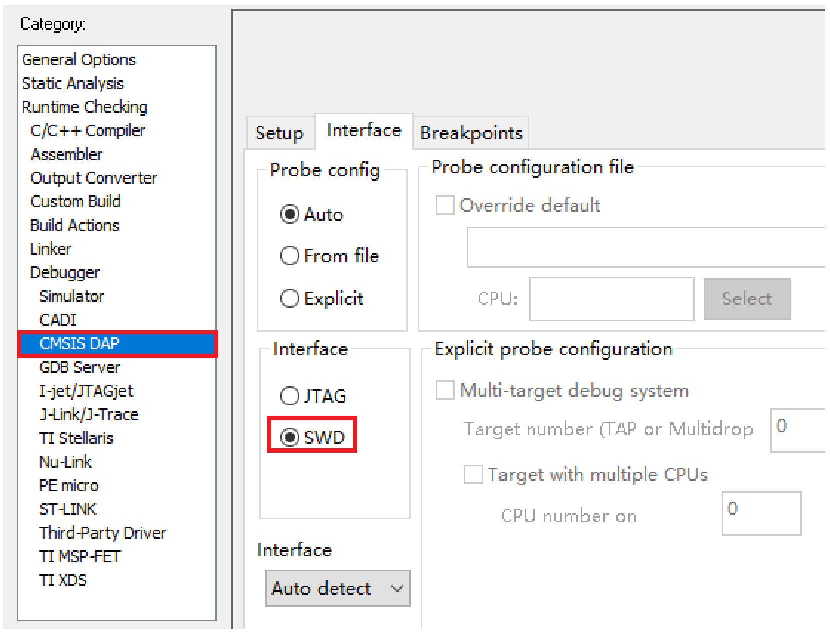


Driver: CMSIS DAP

☑ Run to: main

\debugger\ArteryTek\AT32F407xG_

Figure 15. IAR CMSIS-DAP option


Category:

- General Options
- Static Analysis
- Runtime Checking
- C/C++ Compiler
- Assembler
- Output Converter
- Custom Build
- Build Actions
- Linker
- Debugger
- Simulator
- CADI
- CMSIS DAP
- GDB Server
- I-jet/JTAGjet
- J-Link/J-Trace
- TI Stellaris
- Nu-Link
- PE micro
- ST-LINK
- Third-Party Driver
- TI MSP-FET
- TI XDS

Setup | Interface | Breakpoints

Probe config: [✓ Auto] From file Explicit

Probe configuration file: Override default

CPU: Select

Interface: JTAG [✓ SWD]

Explicit probe configuration: Multi-target debug system

Target number (TAP or Multidrop 0

Target with multiple CPUs

CPU number on 0

Interface: Auto detect

Note: The document "AT32F425firmware_BSP&Pack_user_guide" provides details on Flash algorithms, MCU product series replacement and J-Link. This document is available from ARTERY official website→ Products→Value line→AT32F425 series→BSP (download and unzip, and then go to "\AT32F425_Firmware_Library_Vx.x.x\document").

1.1.4 How to quickly replace AT32F415 with AT32F425

- The migration guide from AT32F415 to AT32F425 is detailed in the document "*MG0019_Migrating from AT32F415 to AT32F425*", which is available from ARTERY official website→Product→Value line→AT32F425 series page.
- If program failure occurs, please refer to the corresponding sections of this document, or you can contact the agent and Artery's technical staff for support.

Note: For more information on how to get better AT32F425 operating performance, refer to the application note "AN0004_Performance_Optimization" from ARTERY official website→Product→AP Note→ AN0004.

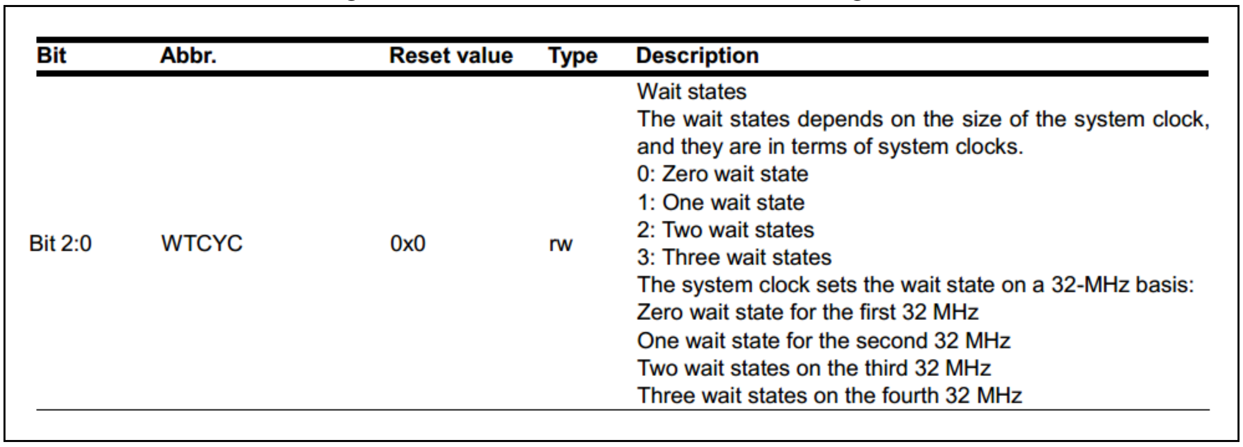
1.2 AT32F425 functionality configuration

1.2.1 Instruction prefetch buffer

Instruction prefetch buffer helps to achieve quicker CPU execution speed. After instruction prefetch buffer is enabled, the subsequent word is already awaiting in the buffer while CPU is reading the current word. The instruction prefetch controller will then determine whether or not to access Flash memory according to available space in the buffer. If there is at least a free space in the instruction prefetch buffer, the instruction prefetch controller will perform a read access.

Different system clocks require different wait states, which can be set through the bit [2:0] (WTCYC) in the FLASH_PSR register.

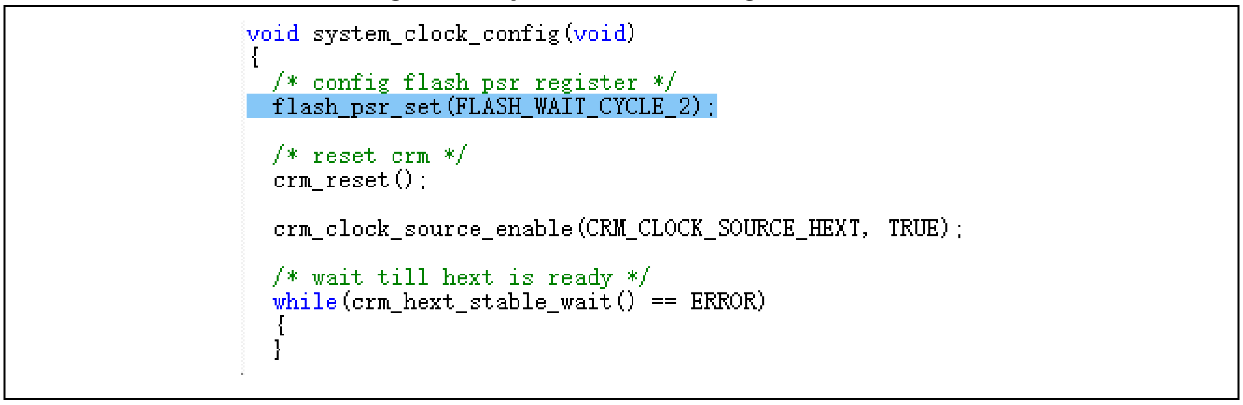
Figure 16. Wait state bit in FLASH_PSR register



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

AT32 library has made relevant settings in the "system_clock_config()" function. For BSP of other AT32 MCU series, you can also find this settings at the same location.

Figure 17. system_clock_config function



```
void system_clock_config(void)
{
  /* config flash psr register */
  flash_psr_set(FLASH_WAIT_CYCLE_2);

  /* reset crm */
  crm_reset();

  crm_clock_source_enable(CRM_CLOCK_SOURCE_HEXT, TRUE);

  /* wait till hext is ready */
  while(crm_hext_stable_wait() == ERROR)
  {
  }
```

1.2.2 PLL clock configuration

AT32F425 embeds a PLL with a maximum of 96 MHz clock output. The AT32F425 PLL clock can be configured by setting the CRM_CFG or CRM_PLL register. The CRM_PLL register can be used to configure different PLL clock frequencies, with the following formula:

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

Example of PLL settings using CRM_CFG register, based on HEXT=8MHz, PLL=96MHz

```
crm_pll_config(CRM_PLL_SOURCE_HEXT, CRM_PLL_MULT_12);
```

Example of PLL settings using CRM_PLL register, based on HEXT=8MHz, PLL=94MHz

Figure 18. AT32F425 94MHz output clock

```
#define  CRM_PLL_NS  ((uint16_t)0x2F)  /* PLL_NS=47 */

#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 "CRM_PLL_SOURCE_HEXT" parameter indicates that the HEXT is used as an external clock source, PLL_NS=47, PLL_MS=1, and PLL_FR value is "CRM_PLL_FR_4 (0x02, divided by 4)".

For details about clock configuration, please refer to *AN0121_AT32F425_CRM_Start_Guide*. This document is available from ATERTY official website→Support→AP Note→AN0121. It introduces how to configure and modify AT32F425 clock source, and how to use the New Clock Configuration tool to quickly generate the desired clock code and apply it to projects.

1.2.3 Encryption

Note: The BOOT1 bit of AT32F425 series is located in the user system data area (0x1FFF F800). When ISP tool is used, it is mandatory to ensure that nBOOT1=1 is asserted (default value) so that the program is booted from system memory instead of SRAM.

1.2.3.1 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.

Note: The high-level access protection, after being enabled, cannot be unlocked. Meanwhile, it is forbidden for users to erase and write system data area in any way.

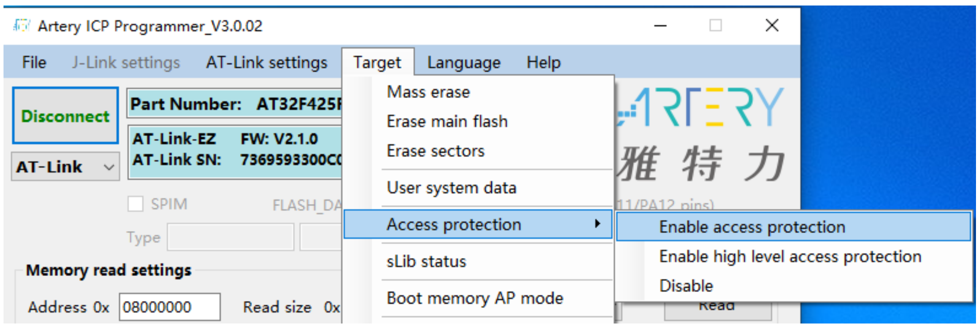
Users can use ICP/ISP programmer to enable or disable access protection.

- Artery ICP Programmer (BOOT0=0)

  Enable access protection: Target – Access protection – Enable access protection or enable high-level access protection (see Figure 19 below).

  Unlock access protection: Target – Access protection – Disable

Figure 19. Use ICP tool to enable/disable access protection



- Artery ISP Programmer (BOOT0=1)

  Enable access protection: Keep clicking "Next" until you enter the final interface. Then tick "Protection" -- choose "Enable" and "Access protection" -- click "Yes" (see Figure 20 below).

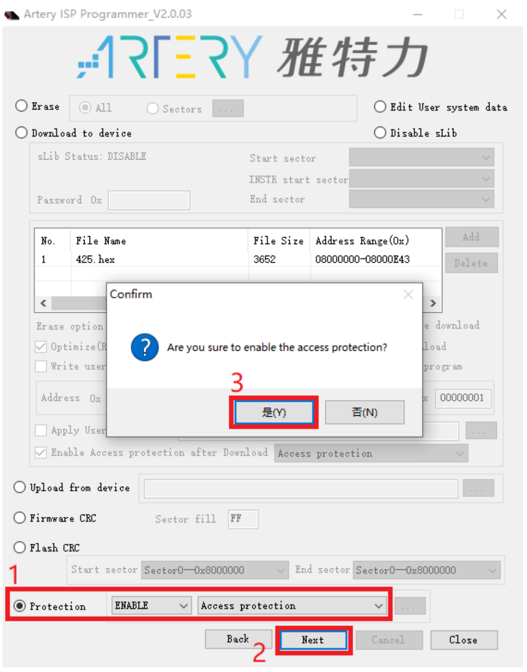
  Unlock access protection: Tick "Protection" -- choose "Disable" and "Access protection" – click "Yes".

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

  Enable access protection: Tick "Protection" -- choose "Enable" and "Access protection" or "High-level access protection" -- click "Start".

Unlock access protection: Tick “Protection” -- choose “Disable” and “Access protection” -- click “Start”.

Figure 20. Use ISP to enable access protection


Artery ISP Programmer_V2.0.03

sLib Status: DISABLE

No.	File Name	File Size	Address Range(0x)
1	425.hex	3652	08000000-08000E43

Confirm

Are you sure to enable the access protection?

是(Y) 否(N)

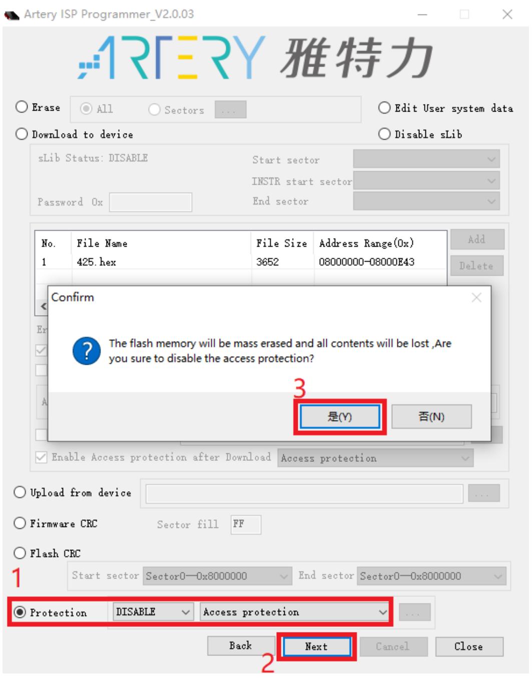
Enable Access protection after Download: Access protection

Sector fill: FF

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

Protection: ENABLE, Access protection

Figure 21. Use ISP to disable access protection


Artery ISP Programmer_V2.0.03

sLib Status: DISABLE

No.	File Name	File Size	Address Range(0x)
1	425.hex	3652	08000000-08000E43

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

1.2.3.2 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)

  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.

- Artery ISP Programmer (BOOT0=1)

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

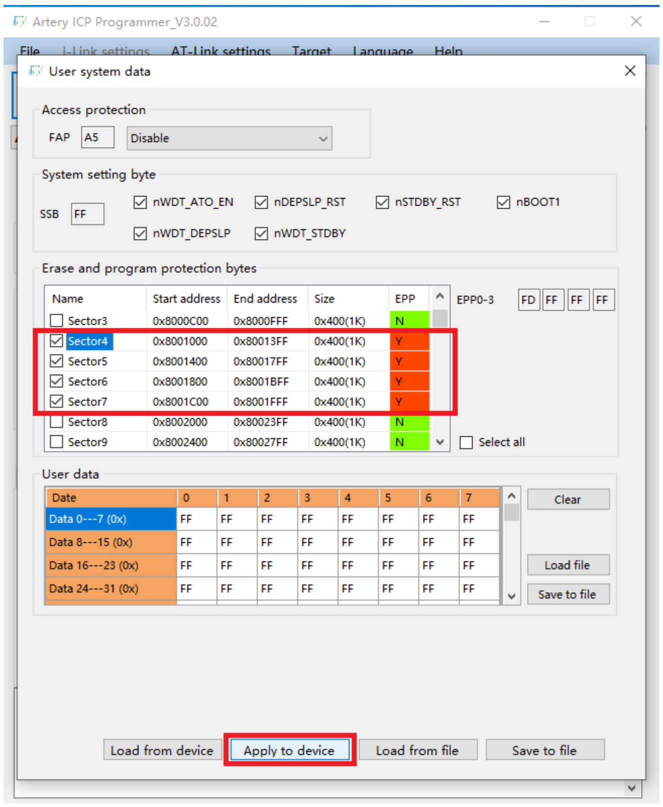
  Disable erase/program protection: Are you sure to disable erase/program protection?--Yes.

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

  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.

Figure 22. Use ICP tool to enable erase/program protection



Access protection

FAP: A5 — Disable

System setting byte

SSB: FF

☑ nWDT_ATO_EN ☑ nDEPSLP_RST ☑ nSTDBY_RST ☑ nBOOT1

☑ nWDT_DEPSLP ☑ nWDT_STDBY

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

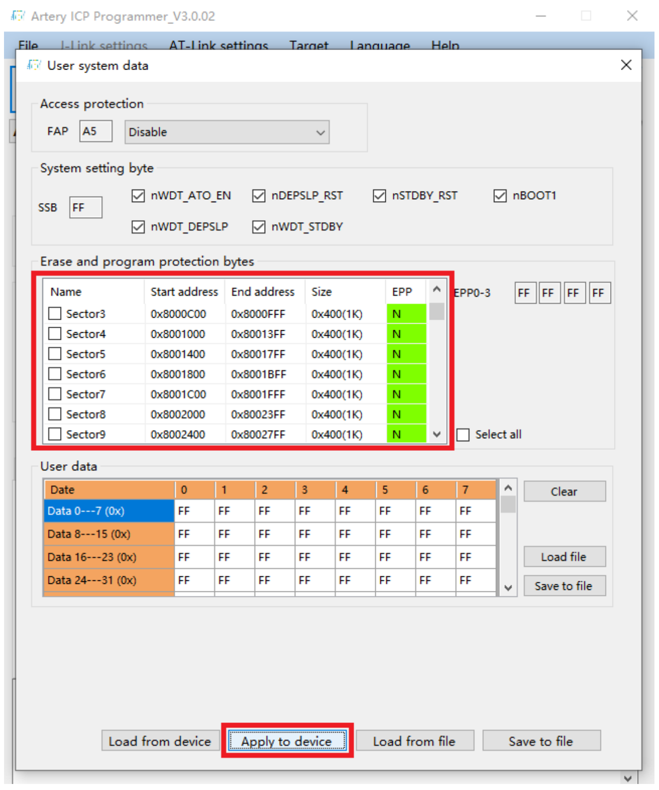
EPP0-3: FD FF FF FF

☐ 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 23. Use ICP tool to disable erase/program protection



Access protection

FAP: A5 — Disable

System setting byte

SSB: FF

☑ nWDT_ATO_EN ☑ nDEPSLP_RST ☑ nSTDBY_RST ☑ nBOOT1
☑ nWDT_DEPSLP ☑ nWDT_STDBY

Erase and program protection bytes

EPP0-3: FF FF FF FF

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

☐ 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: Erase/program protection, after being enabled, cannot be unlocked through erase operation.

1.2.4 Set system memory as main memory extension

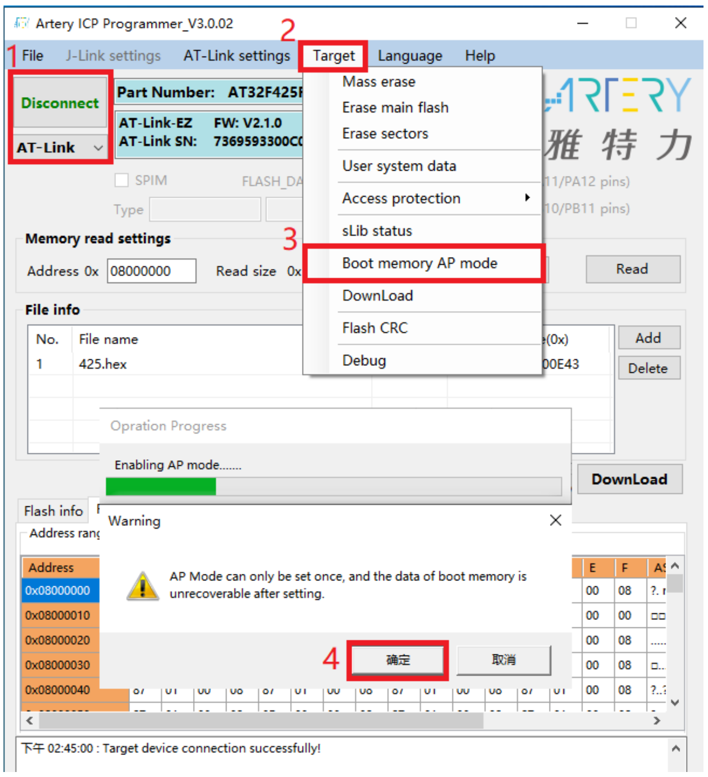
System memory is used as a boot mode by default to store microcontroller manufacturer' Startup Code. On top of this, in AT32F425 series, a new feature is added to the system memory by using it to store user-defined codes as an extended memory area (AP mode).

Note: System memory AP mode is irreversible and can only be set once, meaning that after AP mode is selected, its original BOOT mode cannot be resumed.

During product development, users can use Artery ICP Programmer to enable system memory as an extended memory according to the following procedures.

- Connect AT-Link or J-Link to AT-START-F425 board and supply power;
- Open Artery ICP programmer and choose AT-Link or J-Link connection;
- Go to menu bar: Target – Boot memory AP mode – OK.

Figure 24. Use ICP tool to set boot memory AP mode



Part Number: AT32F425

AT-Link-EZ FW: V2.1.0

AT-Link SN: 7369593300C

Memory read settings

Address 0x 08000000 Read size 0x

File info

No.	File name
1	425.hex

Opration Progress

Enabling AP mode.......

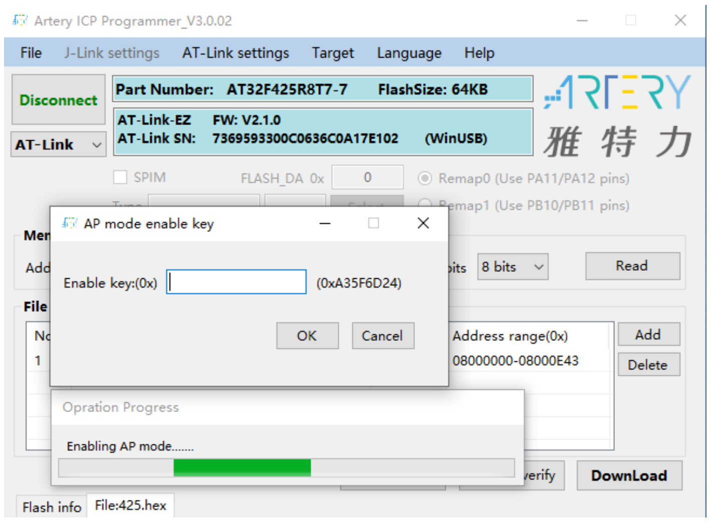
Warning

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

下午 02:45:00 : Target device connection successfully!

■ To avoid unexpected wrong operations, users need manually enter the passkey “0xA35F6D24, and then check whether the operation is successful or not in “File info”.

Figure 25. Boot memory AP mode operating progress



Part Number: AT32F425R8T7-7 FlashSize: 64KB

AT-Link-EZ FW: V2.1.0

AT-Link SN: 7369593300C0636C0A17E102 (WinUSB)

AP mode enable key

Enable key:(0x) (0xA35F6D24)

Address range(0x) 08000000-08000E43

Opration Progress

Enabling AP mode.......

Flash info File:425.hex

In mass production stage, users can use Artery ICP Programmer to enable system memory as memory extension area according to the following procedures:

■ Connect AT-Link to AT-START-F425 board and supply power;

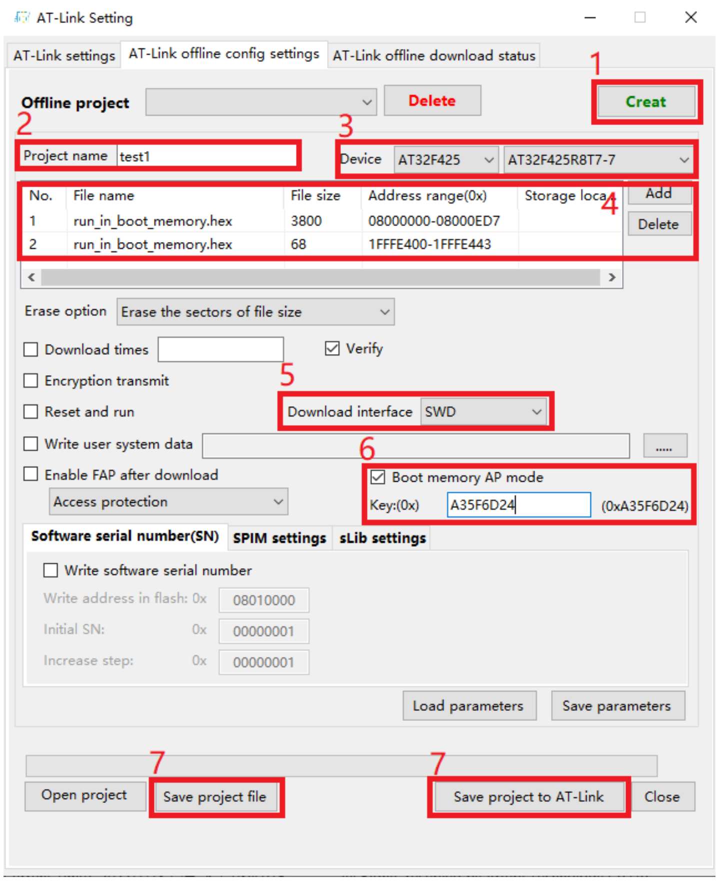
Note: The onboard AT-Link EZ edition of AT-START-F425 does not support offline programming. Therefore, users can only use other editions.

■ Open Artery ICP programmer and choose AT-Link connection;

- Go to menu bar: AT-Link setting -- AT-Link offline configuration setting;
- Follow the steps below to generate off-line project:
  1. Click “Create”
  2. Enter the project name
  3. Select a the desired MCU series and MCU part number
  4. Add *.hex* file
  5. Choose SWD as download interface
  6. Tick “Boot mode AP mode” option and enter the passkey
  7. Save project file or save project to AT-Link

In addition to above settings, users can also make other configurations as needed.

Figure 26. Use ICP tool to set offline boot mode AP mode



Project name: test1

Device: AT32F425 AT32F425R8T7-7

No.	File name	File size	Address range(0x)	Storage loca
1	run_in_boot_memory.hex	3800	08000000-08000ED7	
2	run_in_boot_memory.hex	68	1FFFE400-1FFFE443	

Erase option: Erase the sectors of file size

☐ Download times

☑ Verify

☐ Encryption transmit

☐ Reset and run

Download interface: SWD

☐ Write user system data

☐ Enable FAP after download

Access protection

☑ Boot memory AP mode

Key:(0x) A35F6D24 (0xA35F6D24)

☐ Write software serial number

Write address in flash: 0x 08010000

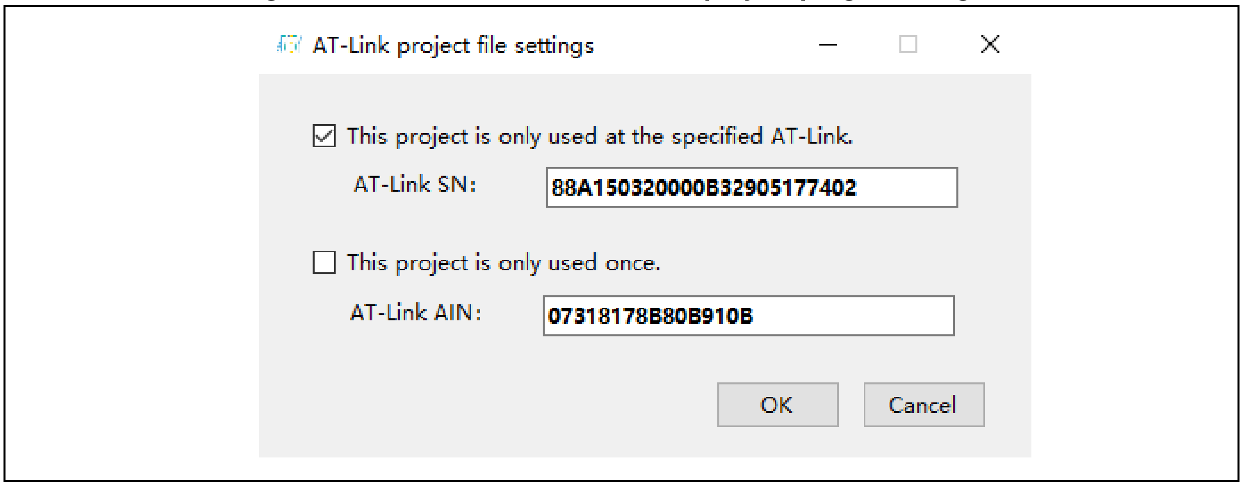
Initial SN: 0x 00000001

Increase step: 0x 00000001

- In Step 7, if you choose “Save project file”, the project will be saved as *.atcp* file so that it can be loaded onto other AT-Link.

  The following dialogue window will pop out during actual operation. If “This project is only used at the specified AT-Link” option is ticked, it means that this project is bonded to a particular AT-Link and can only be used in this AT-Link. In this case, users need to enter AT-Link serial number which will be bonded to. If “This project is only used once” option is ticked, it means that this project could only be used once in the same AT-Link.

Figure 27. Use ICP tool to set offline project programming


AT-Link project file settings

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

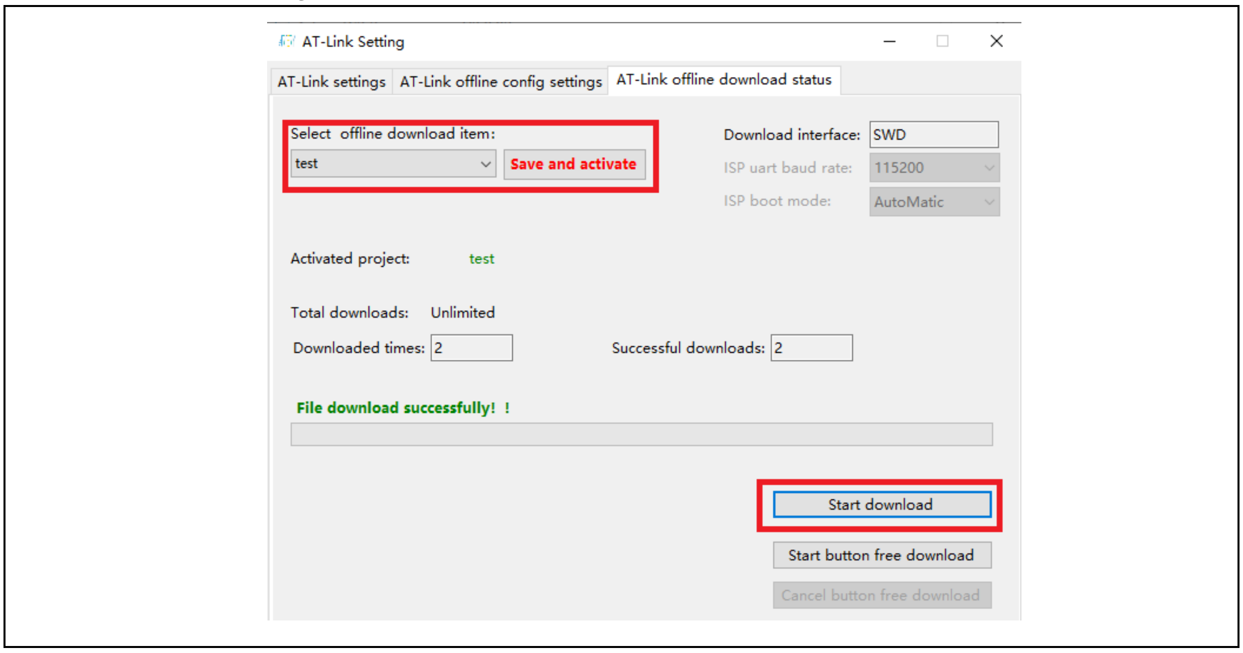
AT-Link SN: 88A150320000B32905177402

☐ This project is only used once.

AT-Link AIN: 07318178B80B910B

- In step 7, if you choose "Save project to AT-Link" and operation is successful, in "AT-Link offline download status" option, you need to choose a project name for offline download; click "Save and activate", and then click "Start download".

Figure 28. ICP tool to monitor offline download status


AT-Link Setting

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

Select offline download item: test

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! !

Start download
Start button free download
Cancel button free download

- For more information on system memory extension, please refer to the document *AN0066_config_boot_memory_as_extension_of_main_memory(AP_mode)*, which is available from ARTERY official website → Support → AP Note → AN0066.
- For demos running user program in the system memory, please refer to the BSP, which is available from ARTERY official website → Product → Value line → AT32F425 series → BSP (download and unzip, and then go to "AT32F425_Firmware_Library_V2.x.x\utilities\at32f425_boot_memory_ap_demo")

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 29. 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 30. Read PID and UID

```
/* 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};

/* AT32F425 MCU type table */
const uint64_t AT32_MCU_ID_TABLE[] =
{
      0x0000010050092100,          //AT32F425R8T7      64KB    LQFP64
      0x0000010050092081,          //AT32F425R6T7      64KB    LQFP64
      …
};

  /* 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*.

2 FAQs about download and compiling

2.1 Program enters Hard Fault Handler

- Access data outside its boundary limit

  Locate where the program exceeds the boundary, and move it to normal data area.
- The SRAM used by the program is outside the programmed MCU SRAM threshold.
- System clock is set out of specification.

2.2 JLink unable to recognize IC in Keil project

For a possible solution, please refer to the following documents:

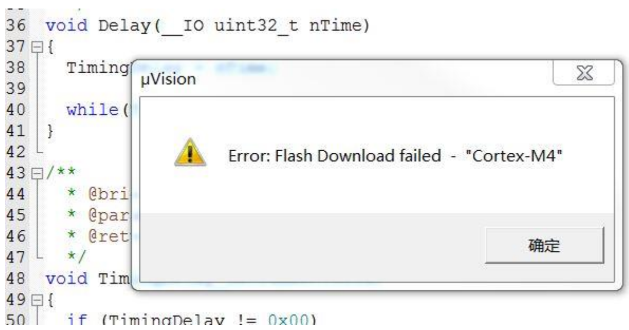
- “*FAQ0008_J-Link_cannot_ find_IC*”, which is available from ARTERY official website →Support→FAQ→FAQ0008.
- “*FAQ0132_How_to_add_Artery_MCU_into_JLINK*”, which is available from ARTERY official website →Support→FAQ→FAQ0132.

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- M4”



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

  while(
}

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

Error: Flash Download failed - "Cortex-M4"

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”.
- Wrong BOOT0 setting. The BOOT0 must be set to 0 to boot MCU from main Flash memory.
- J-Link driver version is incorrect. Versions 6.20C and above are recommended.
- JTAG/SWD PIN 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".
- JTAG/SWD PIN 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:

- JTAG/SWD PIN disabled, so that program download failed and the JTAG/SWD device cannot be found.
- Entered Standby mode 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 Boot0=1, and then press "Reset" button to resume download (note to return to Boot0=0 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 AT32F425 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 or 0x00.
- 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 “*AN0120_AT32F425_Security_Library_Application_Note*” from ARTERY official website → Support → AP Note → AN0120.

4 Revision history

Table 1. Document revision history

Date	Version	Revision note
2022.05.07	2.0.0	Initial release.
2022.07.11	2.0.1	Updated descriptions.
2022.10.11	2.0.2	Updated 3rd party tools and added description of development environment and file paths.
2022.10.21	2.0.3	Updated description of UID and PID.