AN0165
Application Note

Getting Started with AT32F423

Introduction

This document is aimed at helping users with a quick start of developing application solutions based on AT32F423 MCU.

Note: The codes in this document are built around ARTERY's V2.x.x BSP. Attention should be paid to the differences between different versions of BSP when in use.

Applicable products:

Part number	AT32F423xx

Contents

List of tables

List of figures

1 Development resources

Resources download link:

- https://www.arterychip.com/en/index.jsp

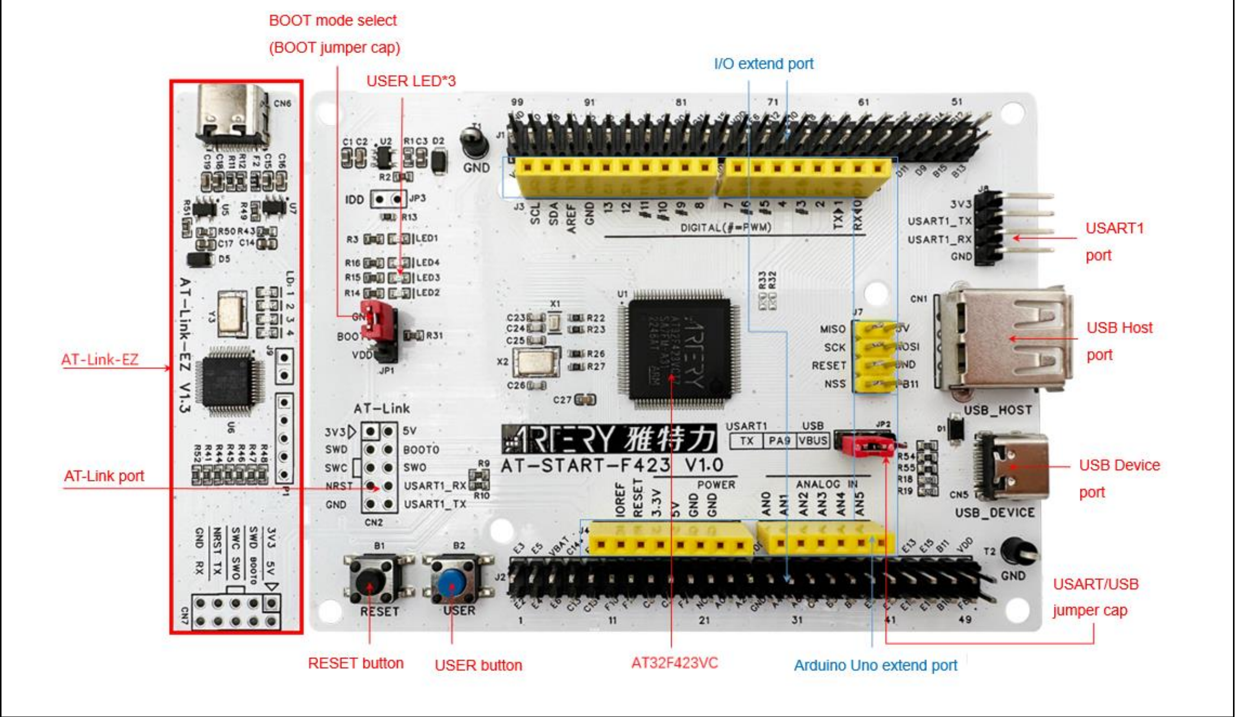
1.1 Set up AT32 development environment

1.1.1 Debug tools and evaluation board

The AT32F423 evaluation board comes with AT-Link-EZ for the debug purpose.

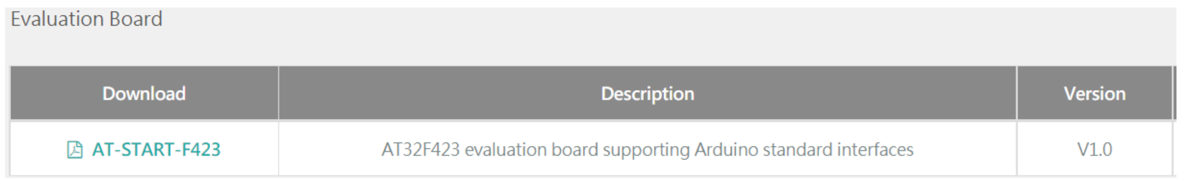
The picture of AT-Link-EZ is marked with red rectangle in Figure 1 below. It can also be separated from the board and works with other circuit boards. The debug tool can be used for several purposes such as IDE online debugging and programming, as well as USB-to-serial interface.

Figure 1. AT-START-F423 and AT-Link-EZ



Note: For details on AT-START-AT32F423 evaluation board, refer to the "UM_AT_START_F423_Vx.x" that is available from ARTERY's official website. You can go to ARTERY official website → PRODUCTS→ Value line → AT32F423 series → Resources → Evaluation Board, where you can download a ZIP-format AT-START-F423 and get \AT_START_F423_Vx.x\03_Documents.

Figure 2. AT-START-F423 evaluation board package from ARTERY official website



Evaluation Board

Download	Description	Version
AT-START-F423	AT32F423 evaluation board supporting Arduino standard interfaces	V1.0

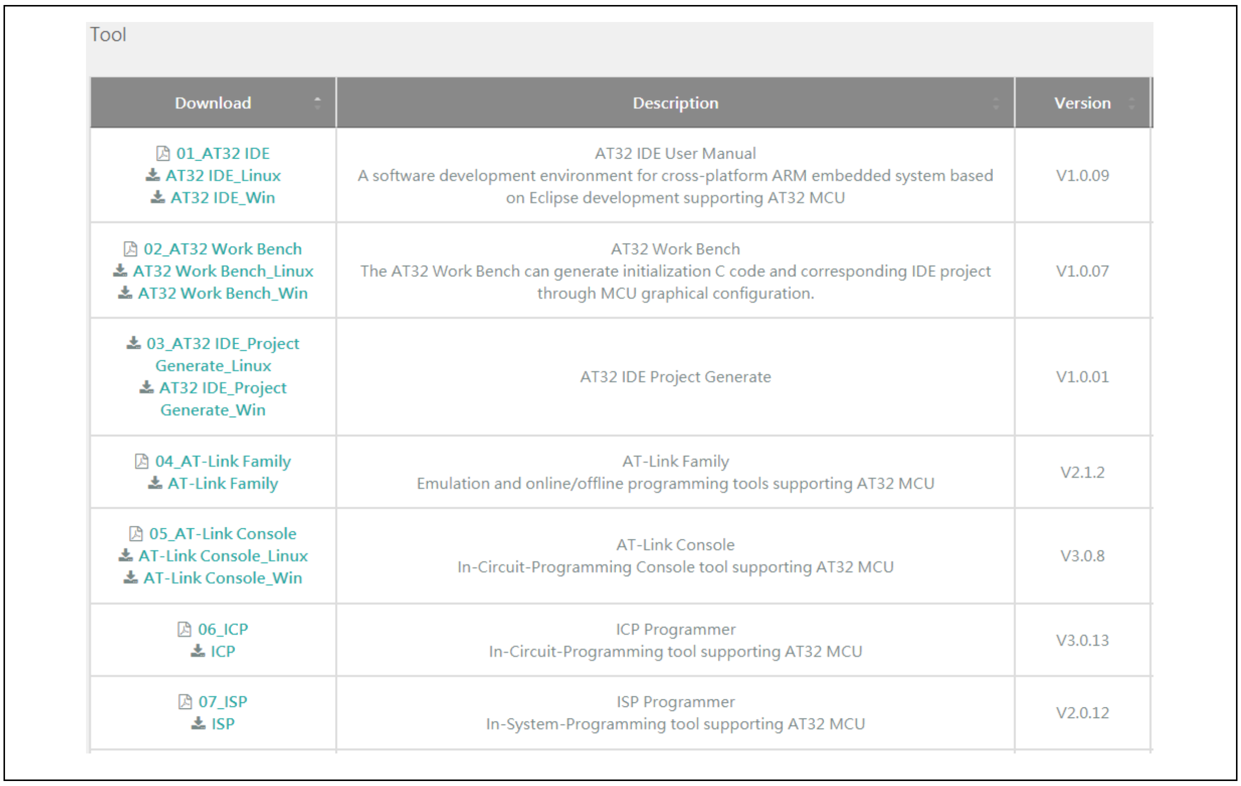
1.1.2 Programming tools and software resources

- **ARTERY programming tools and software:** AT-Link /AT-Link+ /AT-Link-Pro /AT-Link-ISO /AT-Link-EZ, and 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 official website→SUPPORT→Hardware Development Tool and 3RD Party Writer.

- For ICP usage instructions, please refer to *UM_ICP_Programmer*. Path: *ARTERY 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 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 official website*→PRODUCTS→Value line→AT32F4xx; download and unzip AT-Link-Family to get the "AT_Link_EN_Vx.x.x\05_Documents\UM0004_AT-Link_User_Manual_EN_Vx.x.x".

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



Tool

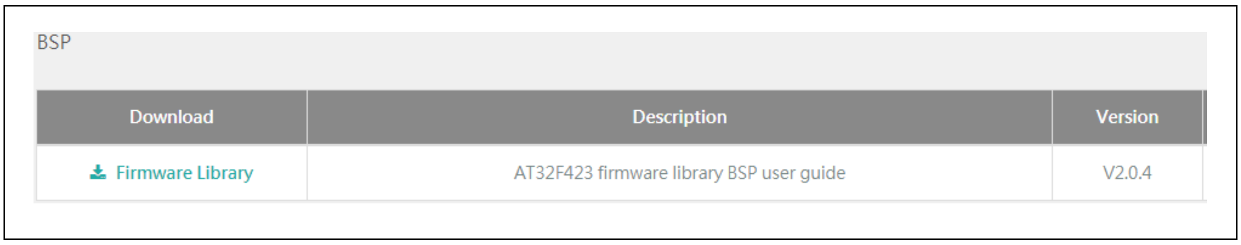
Download	Description	Version
01_AT32 IDE AT32 IDE_Linux AT32 IDE_Win	AT32 IDE User Manual A software development environment for cross-platform ARM embedded system based on Eclipse development supporting AT32 MCU	V1.0.09
02_AT32 Work Bench AT32 Work Bench_Linux AT32 Work Bench_Win	AT32 Work Bench The AT32 Work Bench can generate initialization C code and corresponding IDE project through MCU graphical configuration.	V1.0.07
03_AT32 IDE_Project Generate_Linux AT32 IDE_Project Generate_Win	AT32 IDE Project Generate	V1.0.01
04_AT-Link Family AT-Link Family	AT-Link Family Emulation and online/offline programming tools supporting AT32 MCU	V2.1.2
05_AT-Link Console AT-Link Console_Linux AT-Link Console_Win	AT-Link Console In-Circuit-Programming Console tool supporting AT32 MCU	V3.0.8
06_ICP ICP	ICP Programmer In-Circuit-Programming tool supporting AT32 MCU	V3.0.13
07_ISP ISP	ISP Programmer In-System-Programming tool supporting AT32 MCU	V2.0.12

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→AT32F423 series→BSP.

Figure 4. BSP resources from ARTERY official website

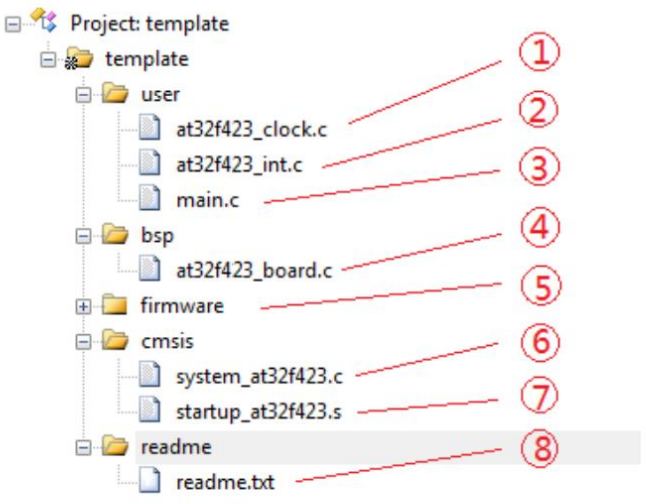


BSP

Download	Description	Version
Firmware Library	AT32F423 firmware library BSP user guide	V2.0.4

BSP offers such template projects as at32_ide/eclipse_gcc/Keil_v5/Keil_v4/IAR_6.10/IAR_7.4/ IAR_8.2/IAR_9.3, which are stored at AT32F423_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 template



Taking Keil_v5 template as an example, it mainly contains the following items:

① **at32F423_clock.c:** it is a clock configuration file defining the default clock frequency and clock path

② **at32F423_int.c:** : it refers to an interrupt file that contains the default process of handling some core interrupts

③ **main.c:** it refers to the main code file

④ **at32F423_board.c:** it refers to the board configuration file that defines common hardware such as buttons and LEDs on AT-START evaluation board

⑤ **firmware:** it contains “at32F423_xx.c” that is used as a driver file for peripherals

⑥ **system_at32F423.c:** system initialization file

⑦ **startup_at32F423.s:** a 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.

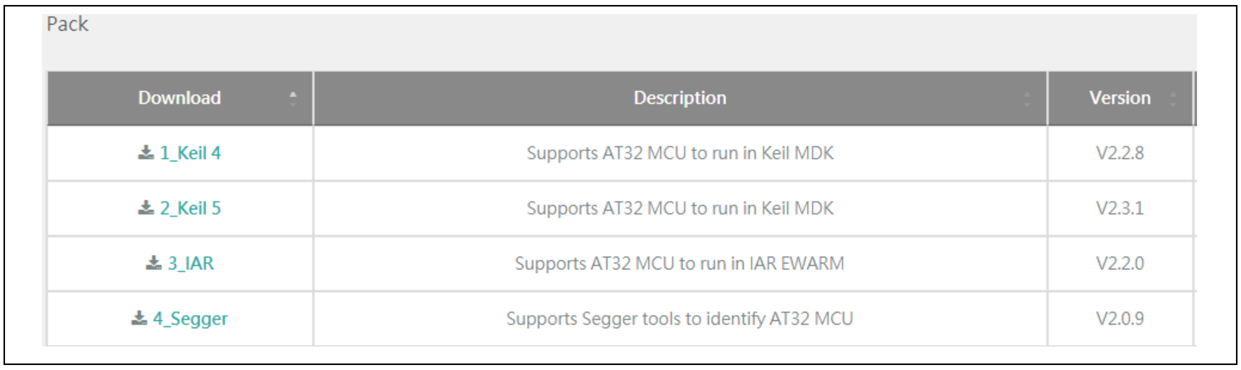
In addition to “Templates”, a large number of examples are included in BSP for users’ reference. These example codes are stored at AT32F423_Firmware_Library_V2.x.x\project\at_start_f4xx \examples.

Note: For details on BSP, refer to Section 4 of the document “AT32F423_firmware_BSP&Pack_user_guide”. This guideline is available from ARTERY official website→PRODUCTS→Value line→AT32F423 series→BSP (unzip and get “\AT32F423_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.
This Pack can be downloaded from *ARTERY official website→PRODUCTS→Value line→AT32* AT32F4xx.

Figure 6. Pack resources from ARTERY official website

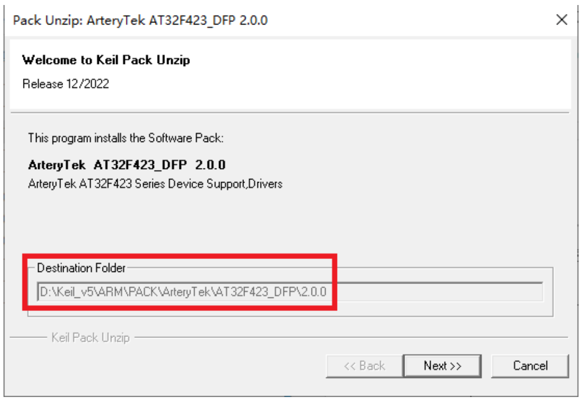


Pack

Download	Description	Version
1_Keil 4	Supports AT32 MCU to run in Keil MDK	V2.2.8
2_Keil 5	Supports AT32 MCU to run in Keil MDK	V2.3.1
3_IAR	Supports AT32 MCU to run in IAR EWARM	V2.2.0
4_Segger	Supports Segger tools to identify AT32 MCU	V2.0.9

Regarding the Keil compiling system, the keil 4.74 or above V5.23 is recommended. For Kei_v5 system, users need first unzip "Keil5_AT32MCU_AddOn" and then install the "ArteryTek.AT32F423_DFP".
For Keil_v4 system, users directly install "Keil4_AT32MCU_AddOn". By default, the installation path of Keil can be automatically recognized when installing. In case of recognition failure, it is necessary for users to manually choose an installation path for Keil.

Figure 7. Install ArteryTek.AT32F423_DFP



Pack Unzip: ArteryTek AT32F423_DFP 2.0.0

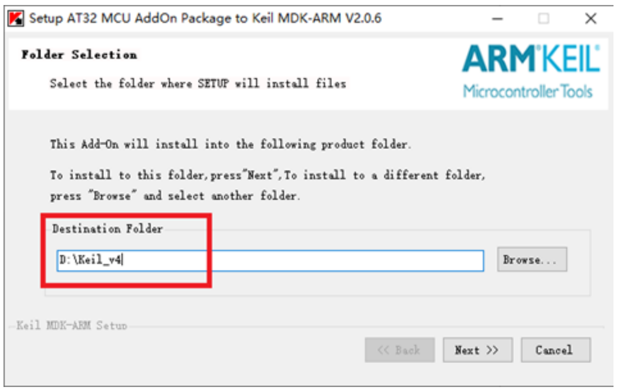
Welcome to Keil Pack Unzip
Release 12/2022

This program installs the Software Pack:

ArteryTek AT32F423_DFP 2.0.0
ArteryTek AT32F423 Series Device Support,Drivers

Destination Folder
D:\Keil_v5\ARM\PACK\ArteryTek\AT32F423_DFP\2.0.0

Figure 8. Install Keil4_AT32MCU_AddOn



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.

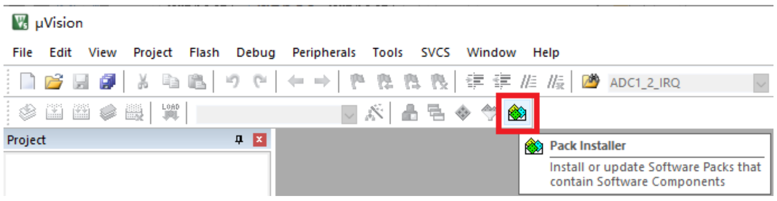
Destination Folder

D:\Keil_v4

Keil MDK-ARM Setup

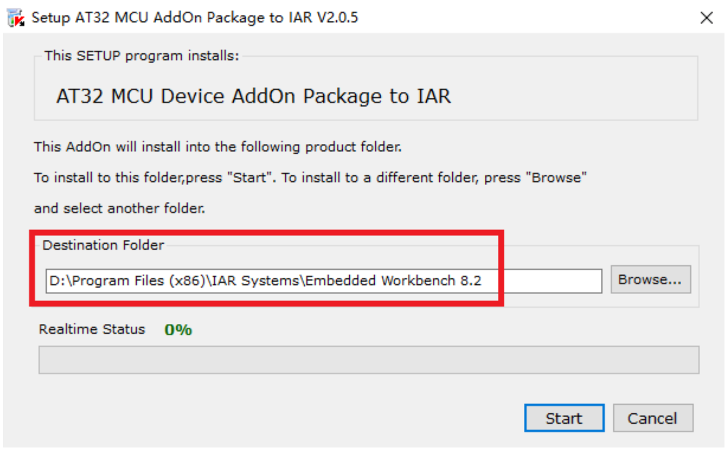
Additionally, you can also open Keil, click the icon "**Pack Installer**", then click "**File**", and choose "**Import**" to import and install "**Pack**" you download from ARTERY official website.

Figure 9. Click "Pack Installer" icon in Keil



If IAR compiling system is to be used, its IAR7.0 or IAR6.1 above versions are recommended. While installing the "IAR_AT32MCU_AddOn", the installation path of Keil can be automatically recognized. In case of recognition failure, it is necessary for users to manually choose an installation path for IAR.

Figure 10. Install IAR_AT32MCU_AddOn



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

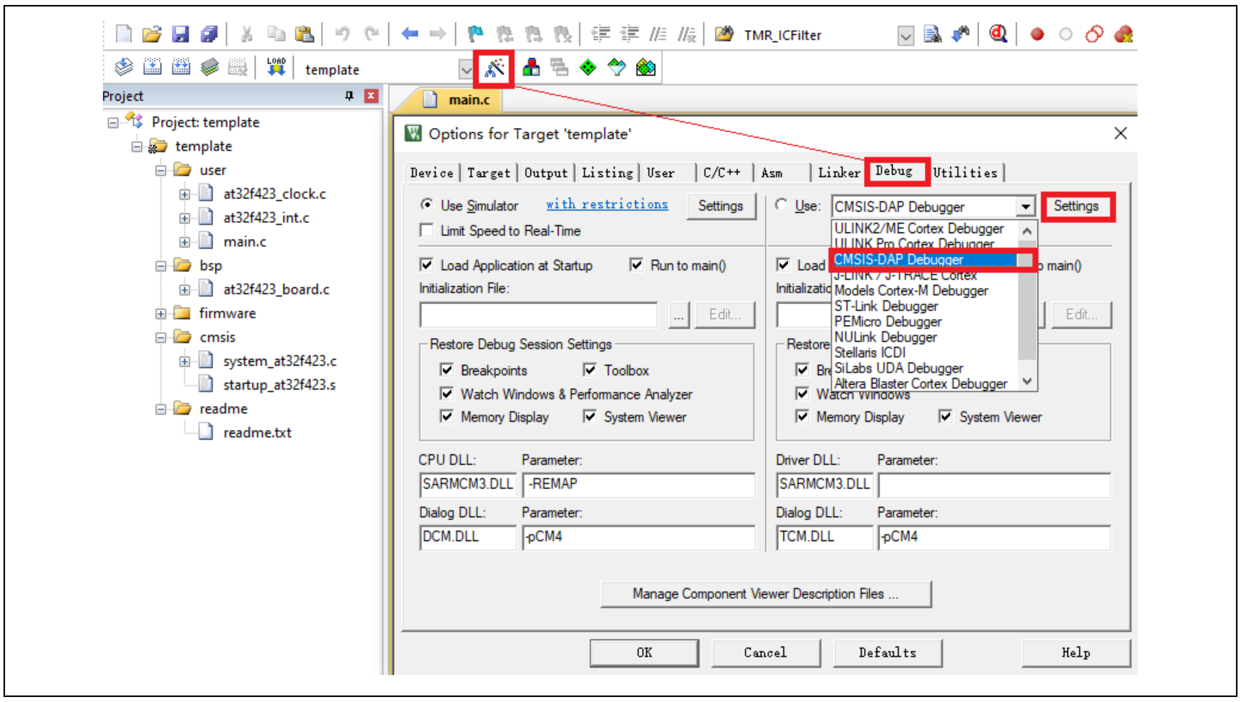
Realtime Status 0%

Note: For details on Pack, refer to Section 2 of the document "AT32F423_firmware_BSP&Pack_user_guide". This guideline is available from [ARTERY official website](#)→PRODUCTS→Value line→AT32F423→BSP (unzip and get the "\AT32F423_Firmware_Library_Vx.x.x\document").

1.1.3.3 Debug and download with AT-Link

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

Figure 11. Keil Debug option



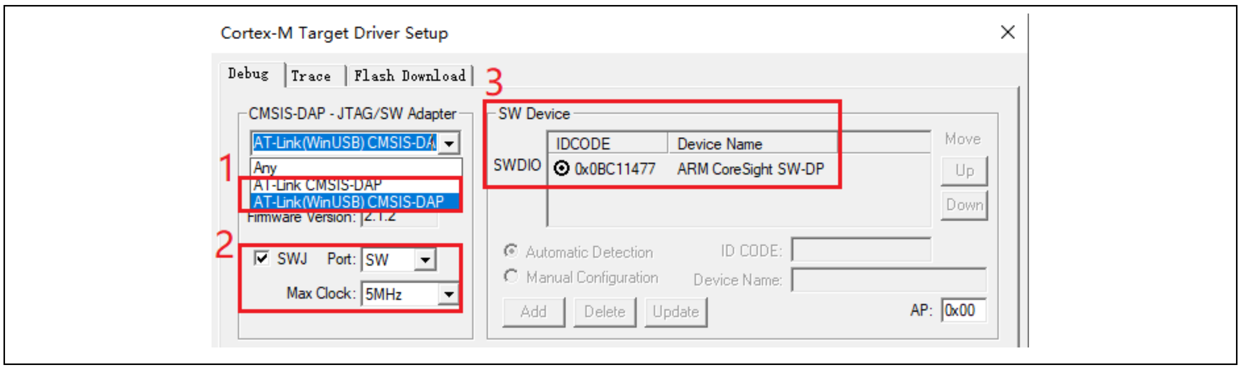
Next, click “**Settings**” to enter “**Cortex-M Target Driver Setup**” window, as shown in Figure 12 below.

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

Note: For more information on WinUSB B, refer to “FAQ0136_How_to_use_AT-LINK_WinUSB_EN_V2.0.0”. Path: ARTERY official website→SUPPORT→FAQ→FAQ0136.

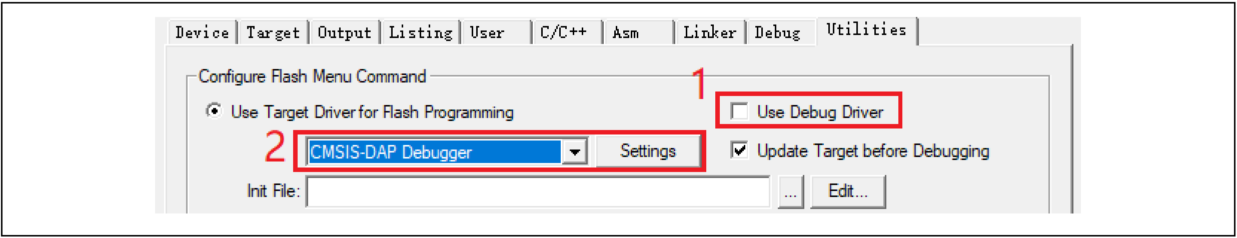
2. In “**Port**” option, select “**SW**” and then tick “**SWJ**”;
3. As shown in step 3 below, the ARM SW-DP debug module is recognized.

Figure 12. Debug Settings in Keil



In “**Utilities**” option, first untick “**Use Debug Driver**” (see step 1), then select “**CMSIS-DAP Debugger**” in Step 2, and finally re-tick Step 1 option (noted that this step 1 must be first unticked before being ticked again later).

Figure 13. Keil Utilities option


Configure Flash Menu Command

(●) Use Target Driver for Flash Programming

CMSIS-DAP Debugger

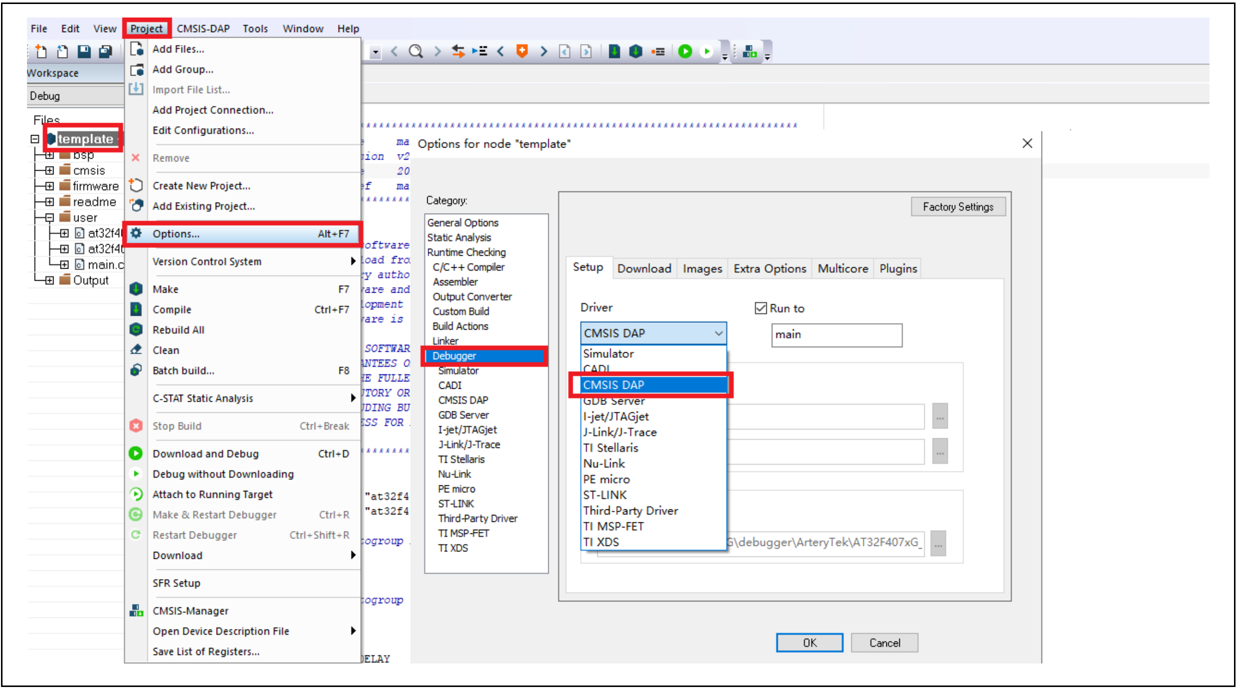
[ ] Use Debug Driver

[✓] Update Target before Debugging

Init File:

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

Figure 14. IAR Debug option


Options for node "template"

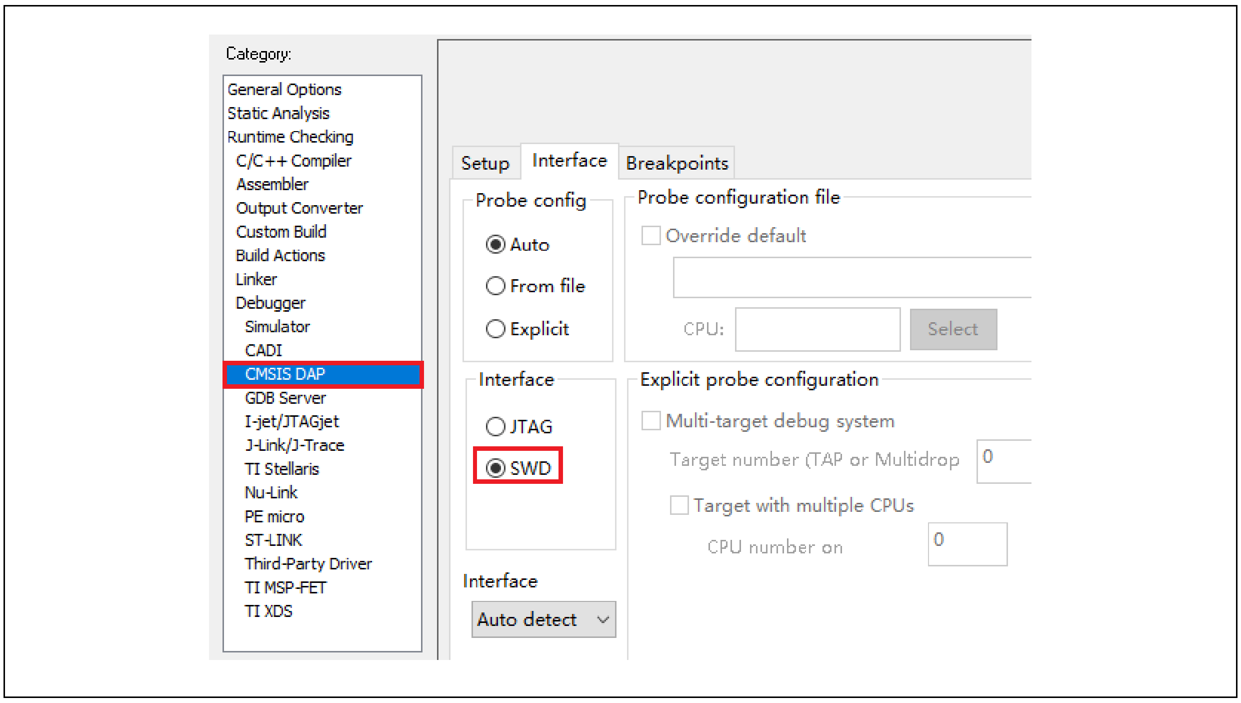
Category: Debugger

Setup

Driver: CMSIS DAP

[✓] Run to: main

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 AT32F423_firmware_BSP&Pack_user_guide provides details on Flash algorithm, MCU product series replacement and J-Link. Thus they are not repeated hereof. This document is located at ARTERY official website→PRODUCTS→Value line→AT32F423→BSP (unzip and get "\AT32F423_Firmware_Library_Vx.x.x\document").

1.1.4 AT32 Work Bench

1.1.4.1 Introduction

This section introduces how to use AT32 Work Bench. Users can use the AT32 Work Bench to graphically configure MC, generate initialization C code and the corresponding IDE project, helping reducing development workload, duration and cost.

- Environmental requirements

  Software:

    Windows: Windows 7 and above

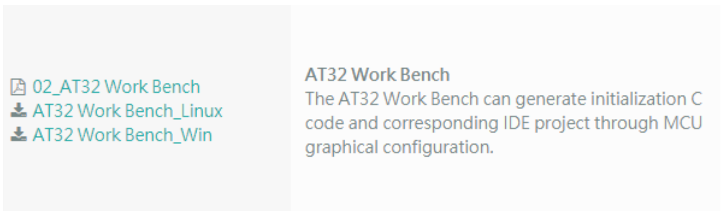
    Linux: Ubuntu or Fedora that support x86_64

  Hardware:

    At least 2GB RAM and 4GB hard drive space

Note: For details about AT32 Work Bench, please refer to UM_AT32_Work_Bench.pdf. Path: ARTERY official website→SUPPORT→Tool. Click https://www.arterychip.com/en/support/index.jsp?index=5, as shown below.

Figure 16. AT32 Work Bench resources from ARTERY official website



AT32 Work Bench
The AT32 Work Bench can generate initialization C code and corresponding IDE project through MCU graphical configuration.

1.1.4.2 Installation

Windows: Run the executable AT32_Work_Bench.exe directly, without the need for installation.

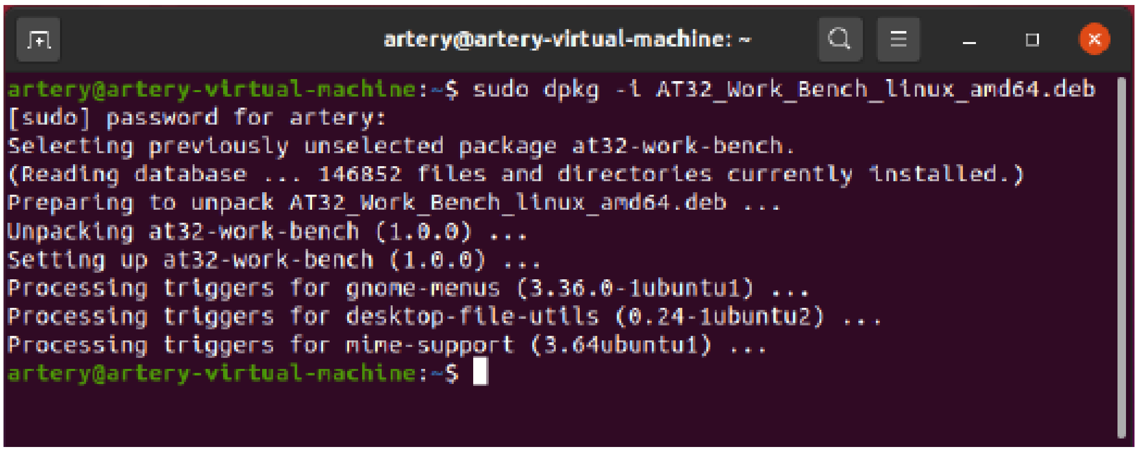
Linux: Ubuntu 16.4 and above are supported.

Installation steps:

■ Enter the following “dpkg” command in the terminal for setup, as shown in Figure 17:

sudo dpgk －i AT32_Work_Bench_Linux-x86_64_Vx.x.xx.deb

Figure 17. dpkg command in Linux

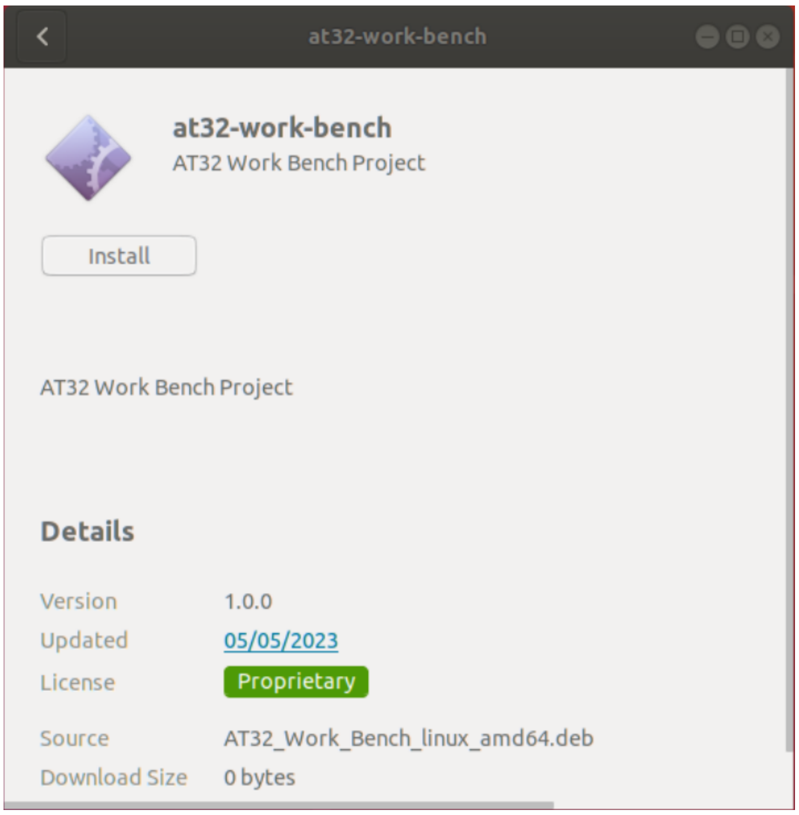


```
artery@artery-virtual-machine:~$ sudo dpkg -i AT32_Work_Bench_linux_amd64.deb
[sudo] password for artery:
Selecting previously unselected package at32-work-bench.
(Reading database ... 146852 files and directories currently installed.)
Preparing to unpack AT32_Work_Bench_linux_amd64.deb ...
Unpacking at32-work-bench (1.0.0) ...
Setting up at32-work-bench (1.0.0) ...
Processing triggers for gnome-menus (3.36.0-1ubuntu1) ...
Processing triggers for desktop-file-utils (0.24-1ubuntu2) ...
Processing triggers for mime-support (3.64ubuntu1) ...
artery@artery-virtual-machine:~$
```

■ Graphical installation

Copy the “AT32_Work_Bench_Linux-x86_64_Vx.x.xx.deb” to Linux and double click. Then, click on the “**Install**”, as shown in Figure 18.

Figure 18. Graphical installation



at32-work-bench

at32-work-bench

AT32 Work Bench Project

Install

AT32 Work Bench Project

Details

Version	1.0.0
Updated	05/05/2023
License	Proprietary
Source	AT32_Work_Bench_linux_amd64.deb
Download Size	0 bytes

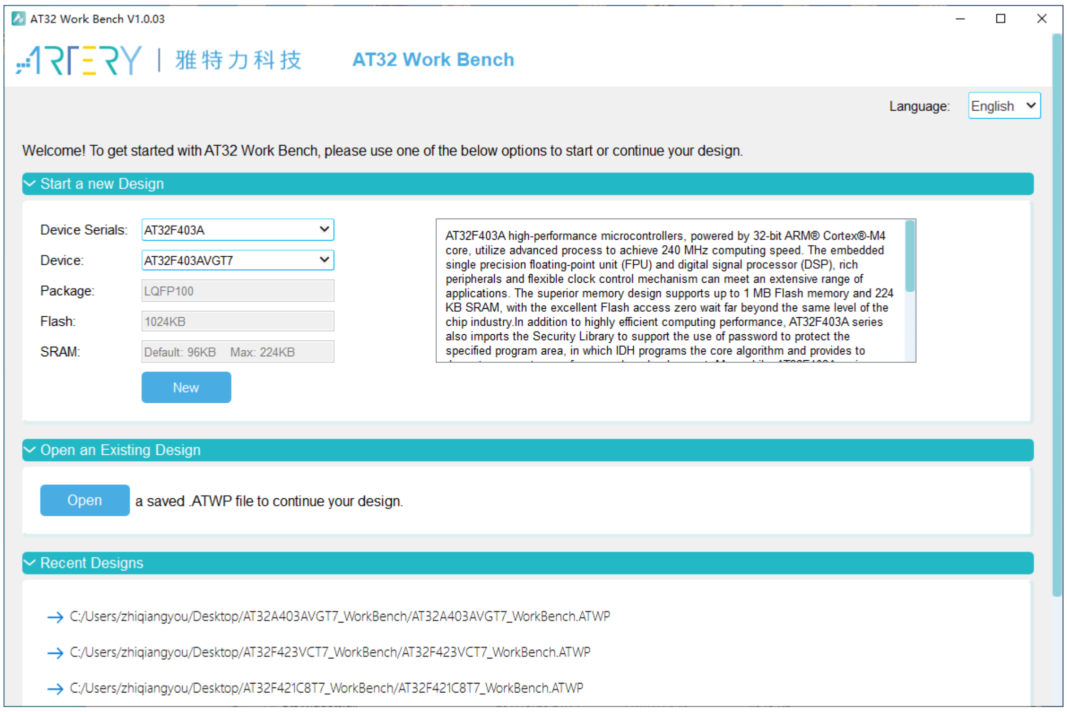
After the installation is complete, click the "**All Programs**" button at the bottom of the left taskbar, find and click on the "AT32 Work Bench" in the program list to start AT32 Work Bench.

1.1.4.3 Project configuration

This section uses AT32F403AVGT7 to create an USART project to show how to use AT32_Work_Bench to generate initialization C code and the corresponding IDE project.

The getting started page is the first window that opens when AT32 Work Bench is started. It includes three options, i.e., “Start a New Design”, “Open an Existing Design” and “Recent Designs”.

Figure 19. AT32 Work Bench getting started page



Welcome! To get started with AT32 Work Bench, please use one of the below options to start or continue your design.

Start a new Design

Device Serials: AT32F403A

Device: AT32F403AVGT7

Package: LQFP100

Flash: 1024KB

SRAM: Default: 96KB Max: 224KB

AT32F403A high-performance microcontrollers, powered by 32-bit ARM® Cortex®-M4 core, utilize advanced process to achieve 240 MHz computing speed. The embedded single precision floating-point unit (FPU) and digital signal processor (DSP), rich peripherals and flexible clock control mechanism can meet an extensive range of applications. The superior memory design supports up to 1 MB Flash memory and 224 KB SRAM, with the excellent Flash access zero wait far beyond the same level of the chip industry.In addition to highly efficient computing performance, AT32F403A series also imports the Security Library to support the use of password to protect the specified program area, in which IDH programs the core algorithm and provides to

Open an Existing Design

Open a saved .ATWP file to continue your design.

Recent Designs

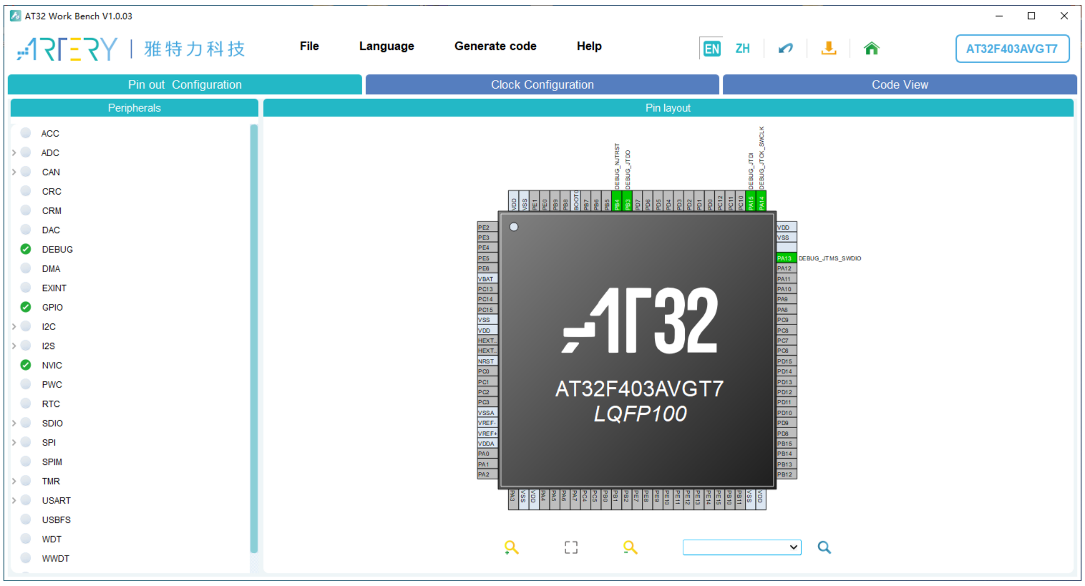
→ C:/Users/zhiqiangyou/Desktop/AT32A403AVGT7_WorkBench/AT32A403AVGT7_WorkBench.ATWP

→ C:/Users/zhiqiangyou/Desktop/AT32F423VCT7_WorkBench/AT32F423VCT7_WorkBench.ATWP

→ C:/Users/zhiqiangyou/Desktop/AT32F421C8T7_WorkBench/AT32F421C8T7_WorkBench.ATWP

Select the desired MCU and click to enter the project configuration window. The project configuration page contains “Pin out configuration”, “Clock configuration” and “Code view”.

Figure 20. Project configuration

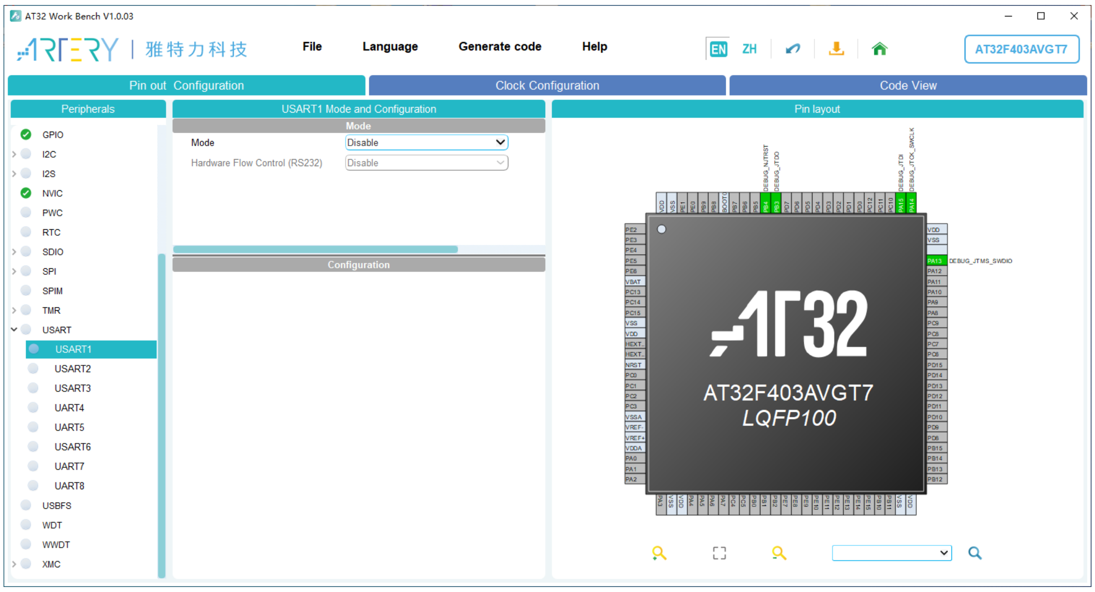


(1) **Pin out configuration:** It is used to configure the MCU peripherals and pins, and it includes “Peripherals”, Mode and configuration” and “Pin layout”.

Select a peripheral to open the “Mode and Configuration” window where users can set the mode and relevant parameters, as well as MCU pins.

One pin is allowed to be used by different peripherals and for multiple functions. Therefore, once the mode is configured, the most suitable peripheral pin layout will be set automatically. For example, Figure 21 shows the “Mode and Configuration” of USART1.

Figure 21. “Mode and Configuration” window



Mode

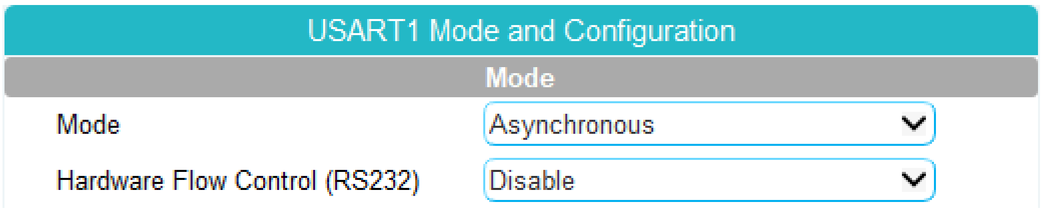
Mode: Disable

Hardware Flow Control (RS232): Disable

Follow the below steps to configure USART1.

■ Peripheral mode

Figure 22. Mode setting



USART1 Mode and Configuration

Mode

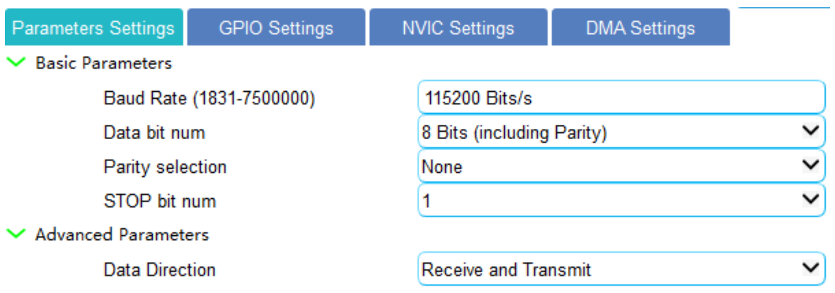
Mode: Asynchronous

Hardware Flow Control (RS232): Disable

As shown in Figure 22, when the USART1 asynchronous mode is selected, PA9 and PA10 are automatically mapped onto “USART1_TX” and “USART1_RX”, respectively.

■ Parameters settings

Figure 23. Parameters settings



Basic Parameters

Baud Rate (1831-7500000): 115200 Bits/s

Data bit num: 8 Bits (including Parity)

Parity selection: None

STOP bit num: 1

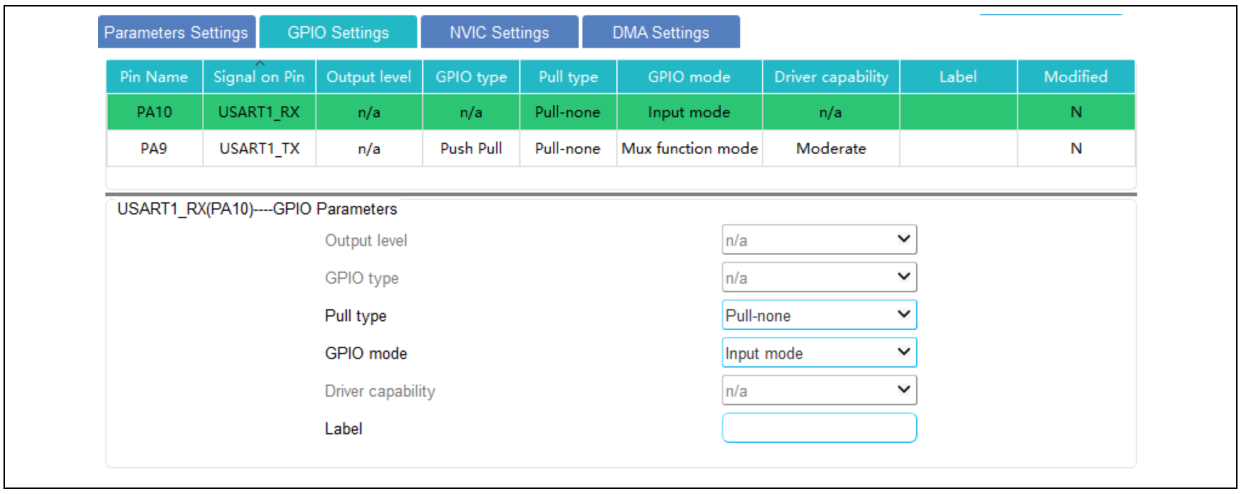
Advanced Parameters

Data Direction: Receive and Transmit

As shown in Figure 23, the “Parameters Settings” window contains USART1 related parameters, including basic parameters such as baud rate and data bit number, and advanced parameters such as data transfer direction.

■ GPIO settings

Figure 24. GPIO settings



Pin Name	Signal on Pin	Output level	GPIO type	Pull type	GPIO mode	Driver capability	Label	Modified
PA10	USART1_RX	n/a	n/a	Pull-none	Input mode	n/a		N
PA9	USART1_TX	n/a	Push Pull	Pull-none	Mux function mode	Moderate		N

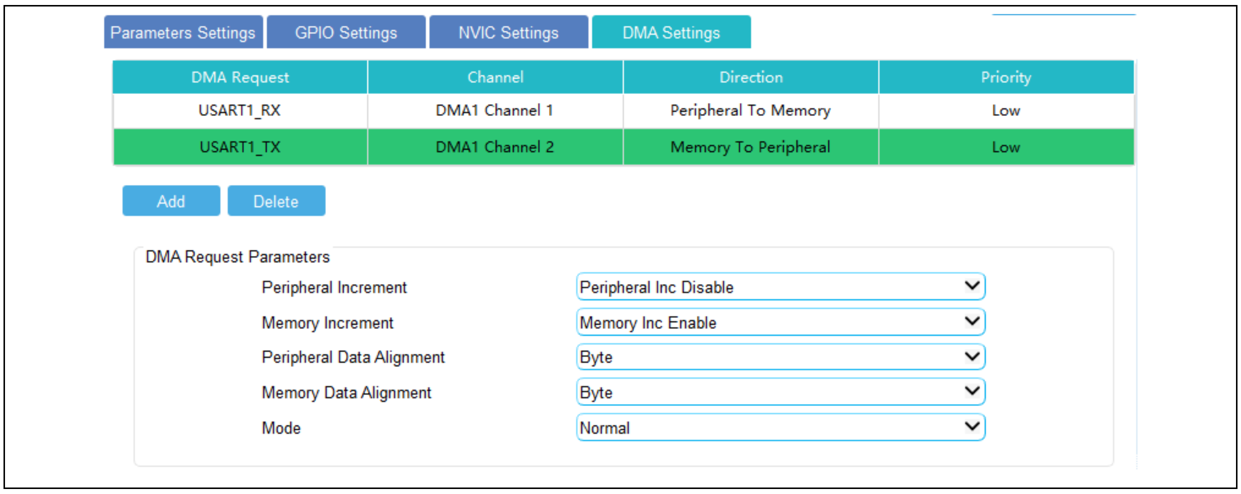
USART1_RX(PA10)----GPIO Parameters

Output level	n/a
GPIO type	n/a
Pull type	Pull-none
GPIO mode	Input mode
Driver capability	n/a
Label	

As shown in Figure 25, the “GPIO Settings” windows contains available GPIOs, such as USART1_TX” and “USART1_RX”.

■ DMA settings

Figure 25. DMA settings



DMA Request	Channel	Direction	Priority
USART1_RX	DMA1 Channel 1	Peripheral To Memory	Low
USART1_TX	DMA1 Channel 2	Memory To Peripheral	Low

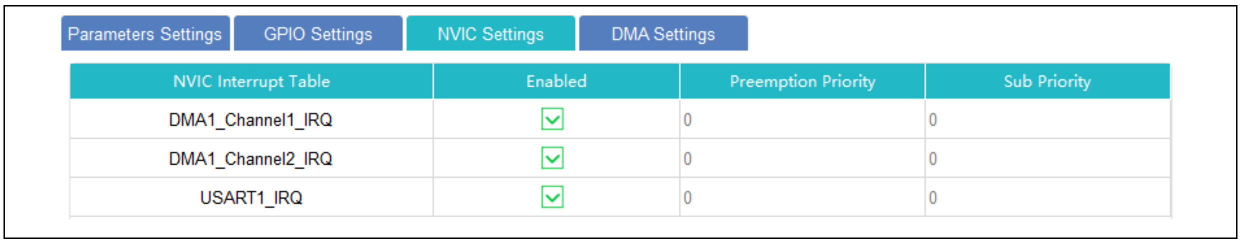
DMA Request Parameters

Peripheral Increment	Peripheral Inc Disable
Memory Increment	Memory Inc Enable
Peripheral Data Alignment	Byte
Memory Data Alignment	Byte
Mode	Normal

As shown in Figure 25, the “DMA Settings” window contains configurable DMA requests, such as DMA channels and DMA request parameters of “USART1_TX” and “USART1_RX”.

■ NVIC settings

Figure 26. NVIC settings



NVIC Interrupt Table	Enabled	Preemption Priority	Sub Priority
DMA1_Channel1_IRQ	✓	0	0
DMA1_Channel2_IRQ	✓	0	0
USART1_IRQ	✓	0	0

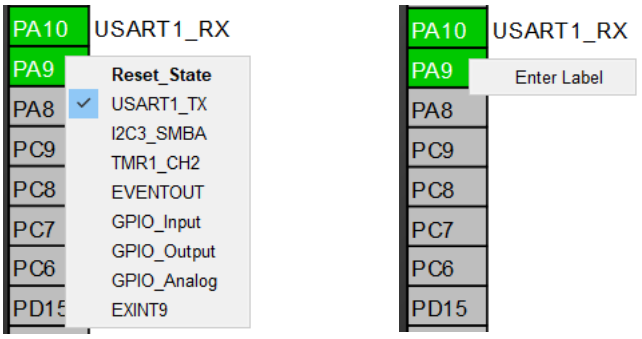
As shown in Figure 26, the “NVIC Settings” window contains configurable interrupts of the selected peripheral. When DMA channel is enabled, the corresponding DMA channel interrupt can be configured. The “preemption priority” and “sub priority” are configured in the “NVIC Mode and configuration” window.

The pin layout of the selected package (such as LQFP48, QFN32 and TSSOP20) is displayed in graphic. Each pin is represented by its name (such as PA9), configuration status, and current signal distribution.

In the pin layout, left click on the pin (except for pins in fixed mode), and then a right-click menu pops up, showing configurable signals for the pin.

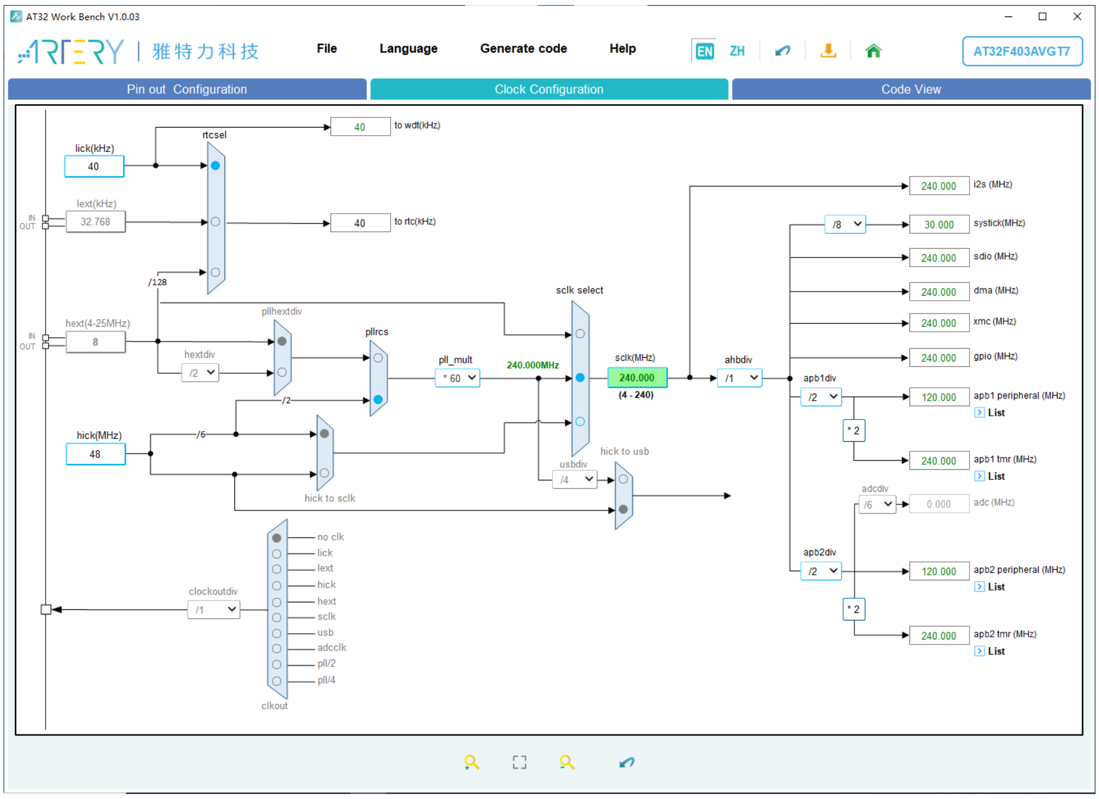
Right click on the configured pin, and then an “Enter Label” button pops up. Users can click on the button to specify a custom label for this signal.

Figure 27. Left click / right click effect



(2) Users can configure the clock path and parameters in the Clock Configuration window, and use drop-down menus and input boxes to modify the actual clock tree configuration to meet application requirements. The Clock Configuration window is shown in Figure 28.

Figure 28. Clock configuration window



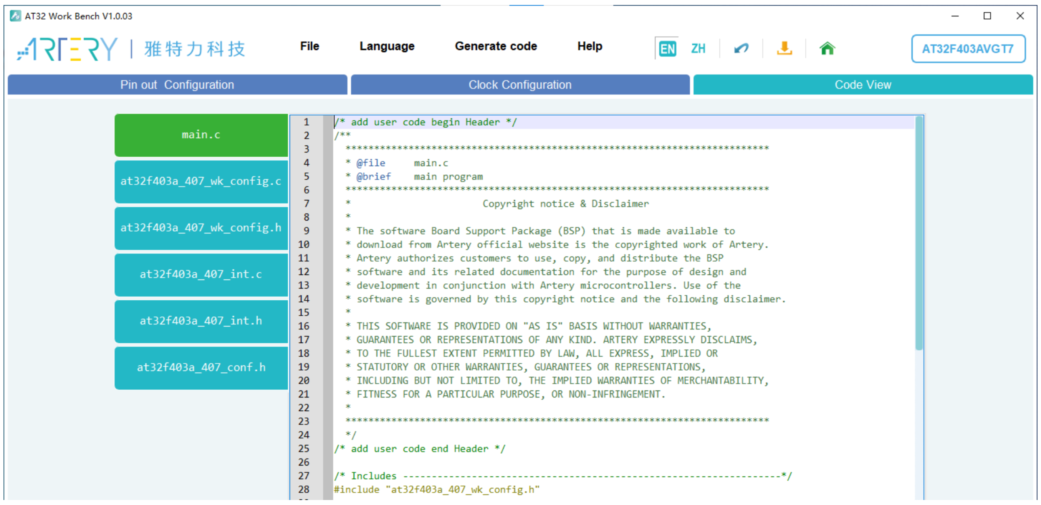
Note 1: Set the “LEXT” mode in the “CRM Mode and Configuration” window to enable LEXT.

Note 2: Set the “HEXT” mode in the “CRM Mode and Configuration” window to enable HEXT.

Note: Tick “Clock Output” in the “CRM Mode and Configuration” window to enable clockout.

(3) Click on the “Code View” to generate code automatically. Code files are listed in the left, and the corresponding code is shown in the right window.

Figure 29. Code view

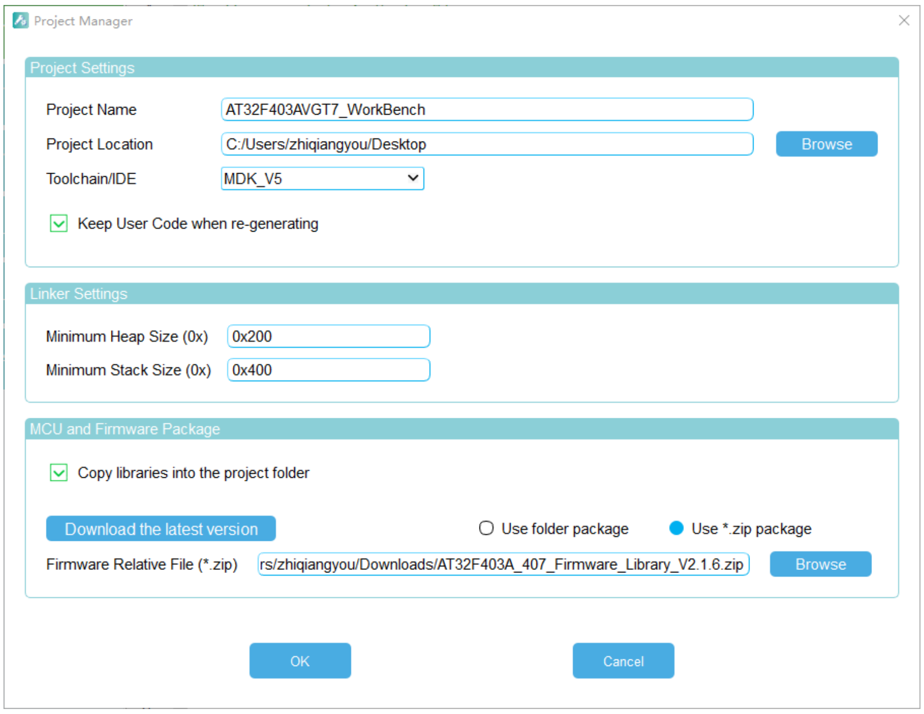


```
/* add user code begin Header */
/**
  **************************************************************************
  * @file     main.c
  * @brief    main program
  **************************************************************************
  *                       Copyright notice & Disclaimer
  *
  * The software Board Support Package (BSP) that is made available to
  * download from Artery official website is the copyrighted work of Artery.
  * Artery authorizes customers to use, copy, and distribute the BSP
  * software and its related documentation for the purpose of design and
  * development in conjunction with Artery microcontrollers. Use of the
  * software is governed by this copyright notice and the following disclaimer.
  *
  * THIS SOFTWARE IS PROVIDED ON "AS IS" BASIS WITHOUT WARRANTIES,
  * GUARANTEES OR REPRESENTATIONS OF ANY KIND. ARTERY EXPRESSLY DISCLAIMS,
  * TO THE FULLEST EXTENT PERMITTED BY LAW, ALL EXPRESS, IMPLIED OR
  * STATUTORY OR OTHER WARRANTIES, GUARANTEES OR REPRESENTATIONS,
  * INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY,
  * FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.
  *
  **************************************************************************
  */
/* add user code end Header */

/* Includes ------------------------------------------------------------------*/
#include "at32f403a_407_wk_config.h"
```

Click on the “Generate code” in the menu bar or toolbar, and then the “Project Manager” window pops up, as shown in Figure 30.

Figure 30. Project manager



Project Name: AT32F403AVGT7_WorkBench

Project Location: C:/Users/zhiqiangyou/Desktop

Toolchain/IDE: MDK_V5

✓ Keep User Code when re-generating

Minimum Heap Size (0x): 0x200

Minimum Stack Size (0x): 0x400

✓ Copy libraries into the project folder

○ Use folder package [✓ ● Use *.zip package]

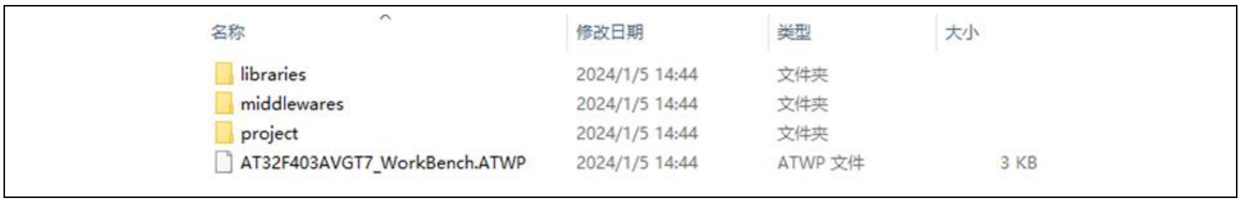
Firmware Relative File (*.zip): rs/zhiqiangyou/Downloads/AT32F403A_407_Firmware_Library_V2.1.6.zip

By default, the minimum heap size and stack size are 0x200 and 0x400, respectively. Users need to modify these values when middleware stack is used.

Tick “Copy libraries into the project folder”, and libraries in the firmware package are copied automatically into the project folder when generating code.

Finally, click on “OK” to generate user code and project of the selected IDE automatically. The generated project file structure is shown in Figure 31.

Figure 31. Project file directory



名称	修改日期	类型	大小
libraries	2024/1/5 14:44	文件夹	
middlewares	2024/1/5 14:44	文件夹	
project	2024/1/5 14:44	文件夹	
AT32F403AVGT7_WorkBench.ATWP	2024/1/5 14:44	ATWP 文件	3 KB

1.1.5 How to quickly replace AT32F415 with AT32F423

- This migration guide from AT32F415 to AT32F423 is detailed in the document “*MG0022_Migrating_from_AT32F415_to_AT32F423_V2.0.0*”, which is available from ARTERY official website→PRODUCTS→Value line→AT32F423 series page.
- If program failure occurs, please refer to the corresponding sections of this document, or you can contact your local or nearest ARTERY Tech team for assistance.

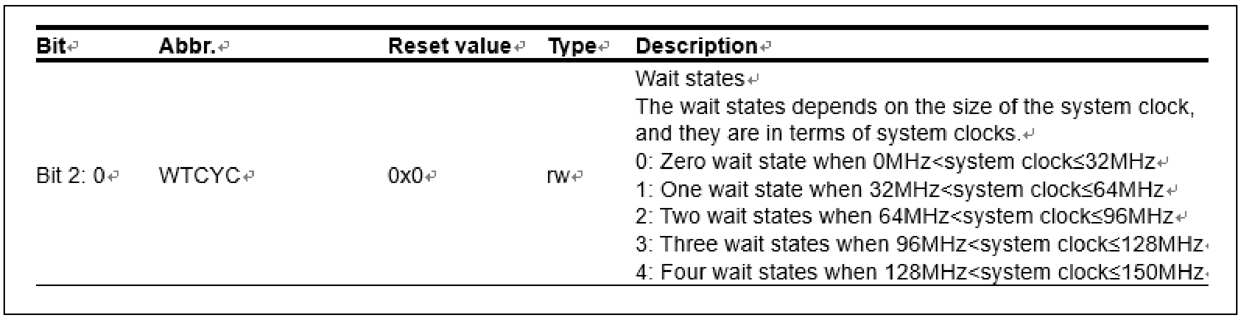
1.2 AT32F423 functionality enhancement

1.2.1 Instruction prefetch buffer

Instruction prefetch buffer is very useful for achieving quicker CPU execution speed. After instruction prefetch buffer is enabled, the subsequent word is already awaiting in the buffer while CPU is reading the existing word. The instruction prefetch controller will then determine whether or not to access Flash memory according to available space in the buffer. Once there is at least a free space in the instruction prefetch buffer, then the instruction prefetch controller will trigger 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.

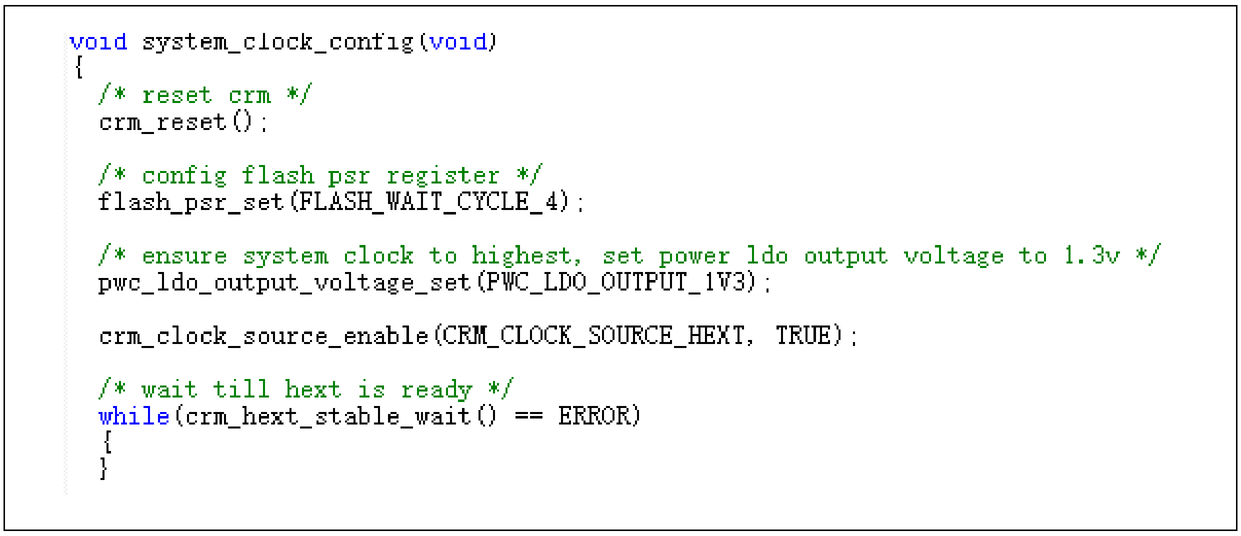
Figure 32. Wait state bit in FLASH_PSR register



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 when 0MHz<system clock≤32MHz 1: One wait state when 32MHz<system clock≤64MHz 2: Two wait states when 64MHz<system clock≤96MHz 3: Three wait states when 96MHz<system clock≤128MHz 4: Four wait states when 128MHz<system clock≤150MHz

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

Figure 33. system_clock_config function



```
void system_clock_config(void)
{
  /* reset crm */
  crm_reset();

  /* config flash psr register */
  flash_psr_set(FLASH_WAIT_CYCLE_4);

  /* ensure system clock to highest, set power ldo output voltage to 1.3v */
  pwc_ldo_output_voltage_set(PWC_LDO_OUTPUT_1V3);

  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 settings

1.2.2.1 PLL settings

AT32F423 embeds a PLL with a maximum of 150 MHz clock output. The CRM_PLLCFG register (PLL clock configuration register) can be used to configure different PLL clock frequencies, with the following formula:

$$\text{PLL output clock} = \left(\text{PLL reference input clock} \times \frac{\text{PLL multiplication frequency factor PLL\_NS}}{\text{PLL pre}-\text{division factor PLL\_MS} \times \text{PLL post}-\text{division factor PLL\_FR}}\right)/2$$

In AT32 BSP, example of PLL settings is based on HEXT=8 MHz, PLL=150 MHz.

Figure 34. AT32F423 150 MHz output clock configuration

```
/* config pll clock resource    PLL_FR =4*/
crm_pll_config(CRM_PLL_SOURCE_HEXT, 150, 1, CRM_PLL_FR_4);
```

Where, the first parameter "**CRM_PLL_SOURCE_HEXT**" represents that the HEXT is used as an external clock source, PLL_NS=**150**, PLL_MS=**1**, and **PLL_FR** value is CRM_PLL_FR_4 (0x02, divided by 4).

For more information on clock configurations, please refer to the document "AN0158_AT32F423_CRM_Start_Guide". Path: ARTERY official website→SUPPORT→AP Note→AN0158. With this file, you can learn more about how to configure and modify AT32F423 clock source code, and how to use ARTERY's New Clock Configuration tool to quickly generate the desired clock code and apply it to project.

The New Clock Configuration document is available from ARTERY official website→PRODUCTS →Value line→AT32F423 series→Tool.

1.2.2.2 PLL automatic frequency switch feature

When the PLL multiplication frequency of AT32F423 series is greater than 108 MHz, it is recommended to enable PLL auto step-by-step frequency switch feature.

Figure 35 gives an example of PLL auto frequency switching function in AT32F423 BSP.

Figure 35. AT32 PLL auto step-by-step frequency switch configuration example

```
/* 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.

1.2.3 Encryption

Note: The BOOT1 bit of AT32F423 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 usually known as an encryption operation. It applies to the entire Flash memory. Once the access protection is enabled, the embedded Flash memory can only be read out through conducting normal program execution, rather than through JTAG or SWD.

Using ISP or ICP tool to unlock access protection will trigger erase operation to the Flash memory.

Note: Once enabled, high-level access protection cannot be unlocked. Meanwhile, it is forbidden for users to erase and write system area in any forms.

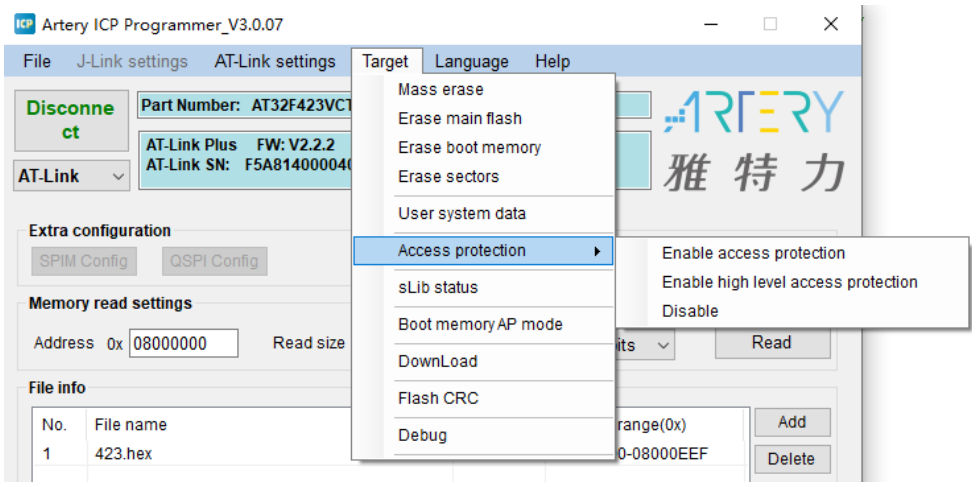
ISP or ISP tool can be used to enable and disable access protection.

- Artery ICP Programmer (BOOT0=0)

  Enable access protection: click Target – Access protection – Enable access protection or enable high-level access protection. (See Figure 36 below)

  Unlock access protection: click Target – Access protection – disable

Figure 36. Use ISP tool to enable or disable access protection



Part Number: AT32F423VCT

AT-Link Plus FW: V2.2.2

AT-Link SN: F5A814000040

Address 0x 08000000

No.	File name	range(0x)
1	423.hex	0-08000EEF

■ Artery ISP Programmer (BOOT0=1)

Enable access protection: Keep clicking “Next” until you enter the final interface. Then check “Protection”, choose “Enable” and “Access protection”, click “Yes” (see Figure 37 below).

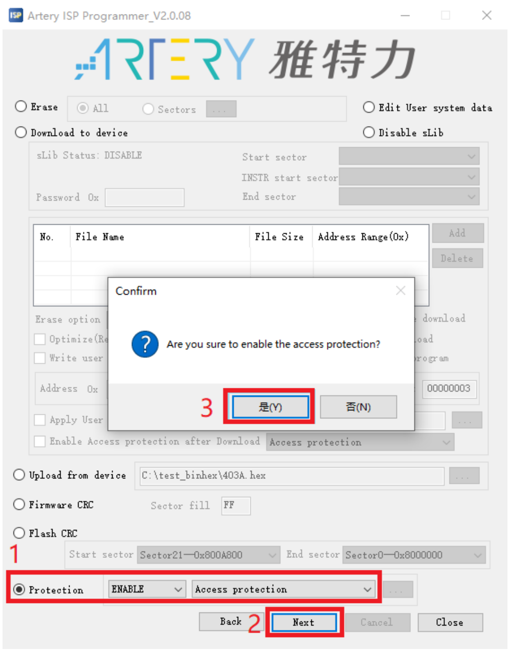
Unlock access protection: check “Protection”, choose “Disable” and “Access protection”, click “Yes”.

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

Enable access protection: check “Protection”, choose “Enable” and “Access protection” or “High-level access protection”, click “Start”.

Unlock access protection: check “Protection”, choose “Disable” and “Access protection”, click “Start”.

Figure 37. Use ISP to enable access protection



sLib Status: DISABLE

No.	File Name	File Size	Address Range(0x)

Are you sure to enable the access protection?

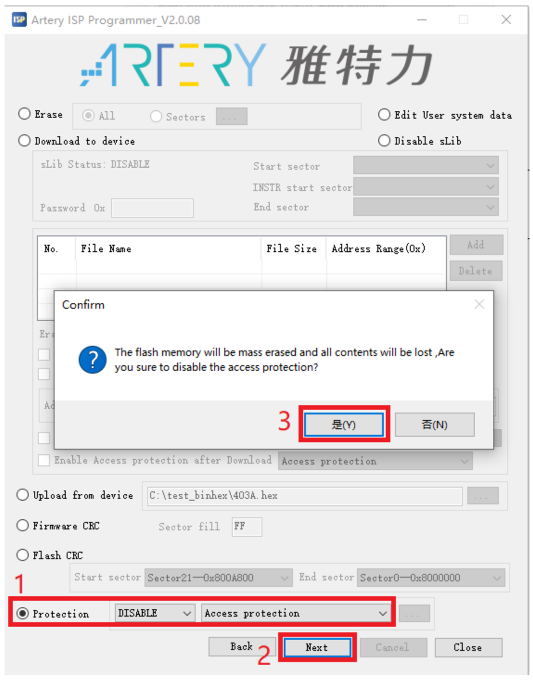
C:\test_binhex\403A.hex

Sector fill FF

Start sector Sector21—0x800A800 End sector Sector0—0x8000000

[✓ Protection] ENABLE Access protection

Figure 38. Use ISP to unlock access protection



sLib Status: DISABLE

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

Access protection

C:\test_binhex\403A.hex

Sector fill FF

Start sector Sector21—0x800A800 End sector Sector0—0x8000000

Protection DISABLE Access protection

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

1.2.3.2 Erase and program protection

Write protection applies to the entire Flash memory or to part of Flash area. Once Flash write protection is enabled, the embedded Flash memory are write-protected against any writing operation.

ISP or ICP tool can be used to enable or disable erase/program protection based on procedure below.

- Artery ICP Programmer (BOOT0=0)

  Enable erase/program protection: click “Target” – User system area – choose the sectors to be erase/write-protected – apply to device

  Disable erase/program protection: click “Target” – User system area – cancel the sectors to be erase/write-protected – apply to device

- Artery ISP Programmer (BOOT0=1)

  Enable erase/program protection: check “Protection” option, choose “Enable” and “Erase/write protection”, and then click “Yes”

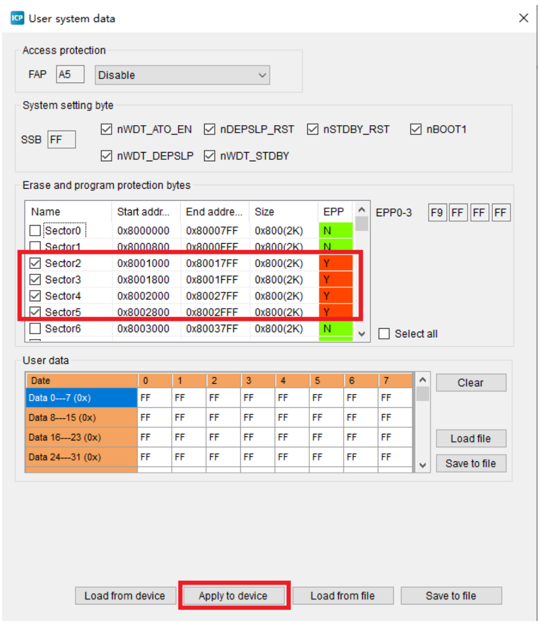
  Disable erase/program protection: check “Protection” option, choose “Disable” and “Erase/ write protection”, and then click “Yes”

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

  Enable erase/write protection: check “Protection” option, choose “Enable” and “Erase/write protection”, and then click “Yes”

  Disable erase/write protection: check “Protection” option, choose “Disable” and “Erase/ write protection”, and then click “Yes”

Figure 39. Use ICP tool to enable erase/program protection


User system data

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 addr...	End addre...	Size	EPP
☐ Sector0	0x8000000	0x80007FF	0x800(2K)	N
☐ Sector1	0x8000800	0x8000FFF	0x800(2K)	N
☑ Sector2	0x8001000	0x80017FF	0x800(2K)	Y
☑ Sector3	0x8001800	0x8001FFF	0x800(2K)	Y
☑ Sector4	0x8002000	0x80027FF	0x800(2K)	Y
☑ Sector5	0x8002800	0x8002FFF	0x800(2K)	Y
☐ Sector6	0x8003000	0x80037FF	0x800(2K)	N

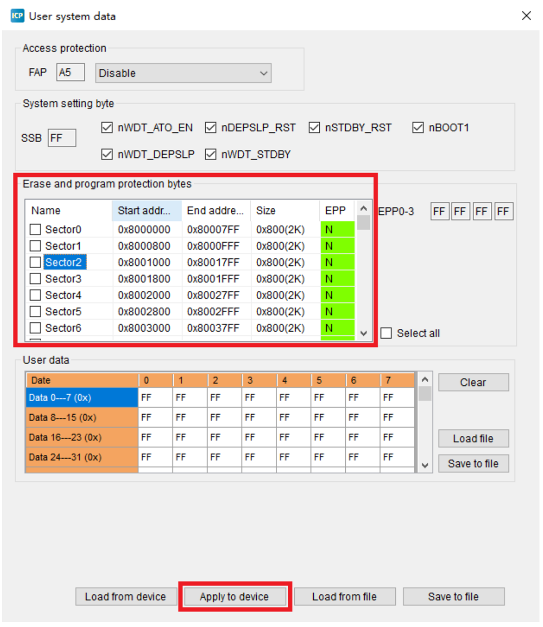
EPP0-3 F9 FF FF FF

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


User system data

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 addr...	End addre...	Size	EPP
☐ Sector0	0x8000000	0x80007FF	0x800(2K)	N
☐ Sector1	0x8000800	0x8000FFF	0x800(2K)	N
☐ Sector2	0x8001000	0x80017FF	0x800(2K)	N
☐ Sector3	0x8001800	0x8001FFF	0x800(2K)	N
☐ Sector4	0x8002000	0x80027FF	0x800(2K)	N
☐ Sector5	0x8002800	0x8002FFF	0x800(2K)	N
☐ Sector6	0x8003000	0x80037FF	0x800(2K)	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: Erase/program protection, after enabled, cannot be unlocked through erase operation.

1.2.4 Set system memory as main memory extension

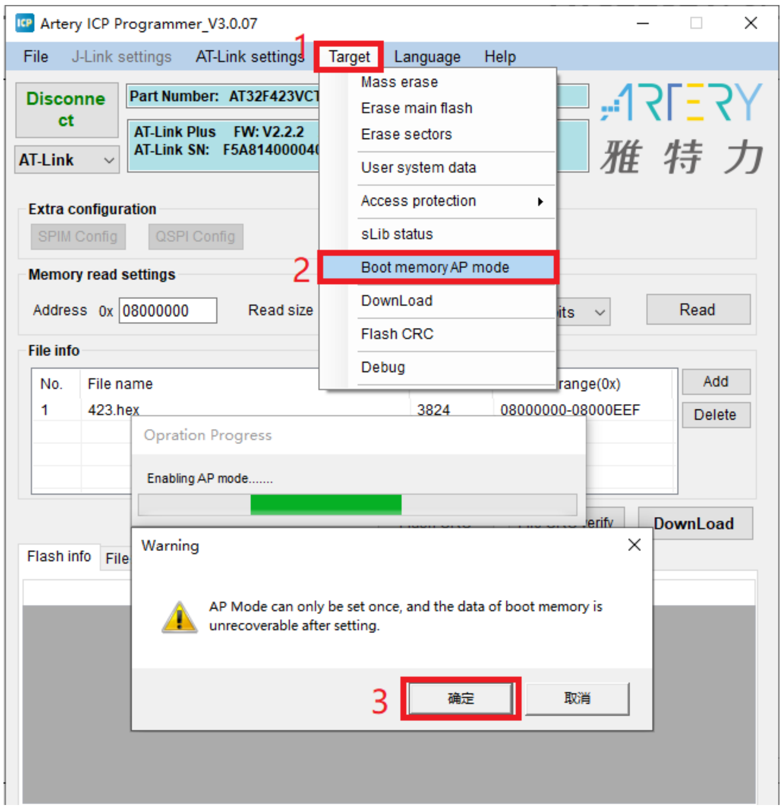
System memory is used as a boot mode by default to store microcontroller manufacturer' Startup Code. On top of this, in AT32F423 series, a new feature is added to the system memory by using it to store user-defined codes as an extended memory area (AP mode).

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

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

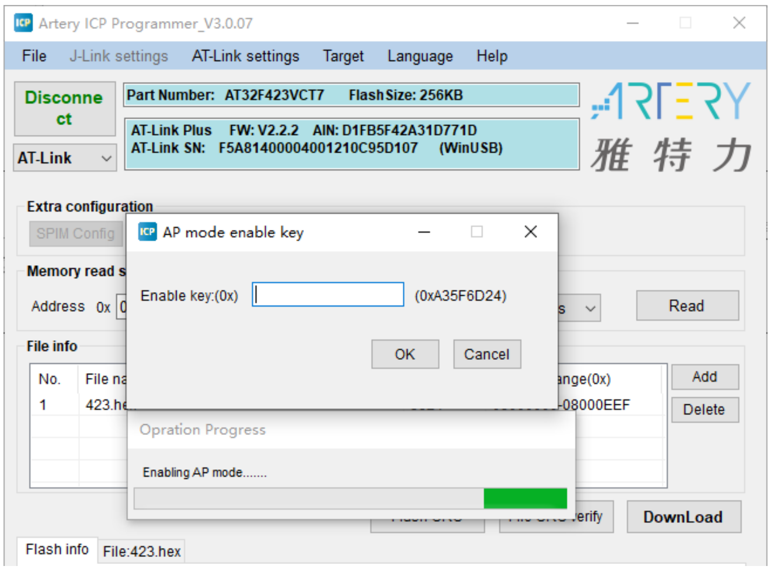
- Connect AT-Link/J-Link to AT-START-F423 evaluation board and supply power to it;
- Open Artery ICP programmer and choose AT-Link/J-Link connection;
- Go to menu bar: Target – Boot memory AP mode – OK.

Figure 41. Use ICP tool to set boot memory AP mode



- To prevent unexpected wrong operations, you need manually enter the passkey “0xA35F6D24”, then you can check whether the operation is successful or not in “File info” column.

Figure 42. Boot memory AP mode operating progress



Part Number: AT32F423VCT7 Flash Size: 256KB

AT-Link Plus FW: V2.2.2 AIN: D1FB5F42A31D771D
AT-Link SN: F5A81400004001210C95D107 (WinUSB)

Enable key:(0x) (0xA35F6D24)

Enabling AP mode.......

During mass-production stage, we can also use Artery ICP Programmer to enable system memory as memory extension area according to the following procedures:

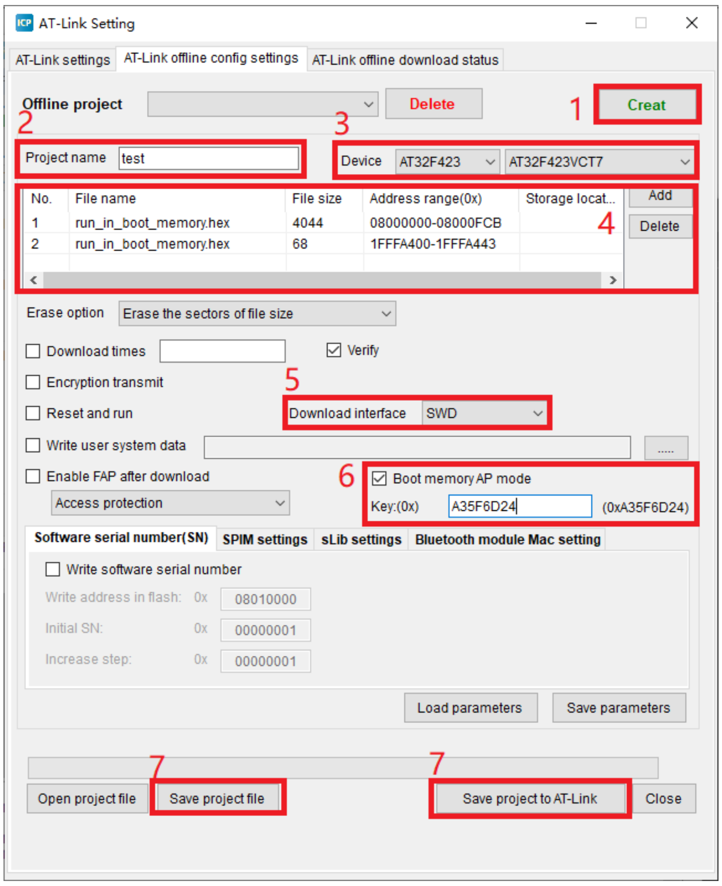
- Connect AT-Link to AT-START-F423 evaluation board and supply power to it.

Note: The on-board AT-Link EZ does not support offline programming. As a result, you can only use non-EZ AT-Link.

- 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 an offline project:
  1. Click “Create”;
  2. Enter a project name;
  3. Select a particular MCU series and MCU part number;
  4. Add .hex files;
  5. Select SWD as download interface;
  6. Tick “Boot memory AP mode” option and enter the passkey;
  7. Save project file or save project to AT-Link.

In addition to above settings, you can also make other configurations according to the actual needs.

Figure 43. Use ICP tool to set offline boot memory AP mode


AT-Link Setting

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

Offline project: (empty) — Delete — Creat

Project name: test

Device: AT32F423 / AT32F423VCT7

No.	File name	File size	Address range(0x)	Storage locat...
1	run_in_boot_memory.hex	4044	08000000-08000FCB	
2	run_in_boot_memory.hex	68	1FFFA400-1FFFA443	

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)

Software serial number(SN) | SPIM settings | sLib settings | Bluetooth module Mac setting

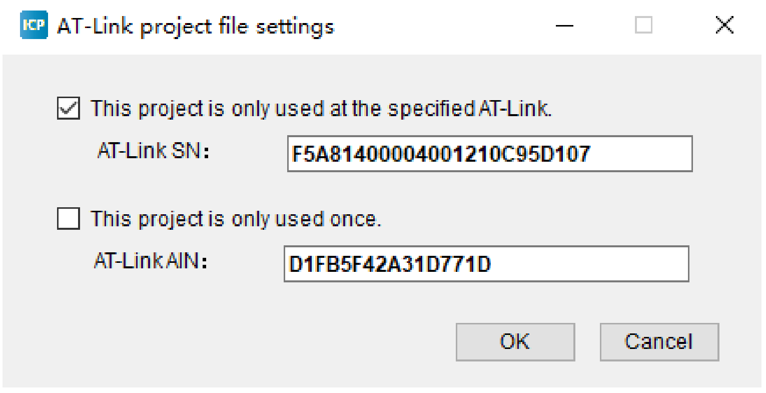
☐ Write software serial number
Write address in flash: 0x 08010000
Initial SN: 0x 00000001
Increase step: 0x 00000001

- In Step 7, if you choose “Save project file”, this 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 checked, it means that this project is bonded to a particular AT-Link and can only be used in this AT-Link. In this case, you need to enter AT-Link serial number that will be bonded to.

  If “This project is only used once” option is checked, it means that this project could only be used once in the same AT-Link.

Figure 44. Use ICP tool to set offline project programming


AT-Link project file settings

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

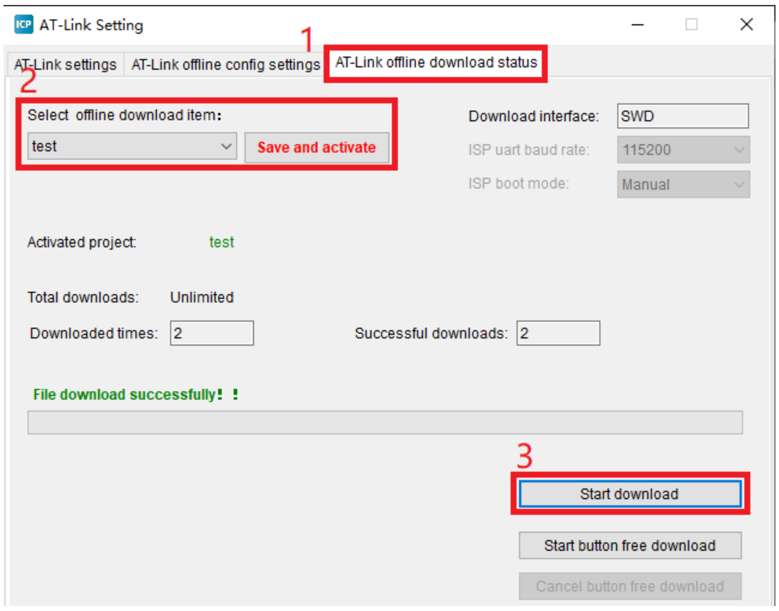
AT-Link SN: F5A81400004001210C95D107

☐ This project is only used once.

AT-Link AIN: D1FB5F42A31D771D

- 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 click “Start download”.

Figure 45. ICP tool to monitor offline download status


AT-Link Setting
AT-Link settings
AT-Link offline config settings
AT-Link offline download status
1
2
3
Save and activate
Start download
Start button free download
Cancel button free download

Select offline download item: test

Download interface: SWD

ISP uart baud rate: 115200

ISP boot mode: Manual

Activated project: test

Total downloads: Unlimited

Downloaded times: 2

Successful downloads: 2

File download successfully! !

- 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 stored at ARTERY official website→SUPPORT→AP Note→AN0066.
- For DEMO on running user program in the system memory, please refer to BSP, which is stored at ARTERY official website→PRODUCTS→Value line→AT32F423 series→BSP (unzip and get the AT32F423_Firmware_Library_V2.x.x\utilities\at32F423_boot_memory_ap_demo).

1.2.5 How to distinguish AT32 MCU from other MCUs

■ **Reading Cortex-M series CPU ID number, you can determine whether it is based on M0, M0+, M1, M3 or M4 core.**

Figure 46. Read Cortex ID

```
cortex_id = *(uint32_t *)0xE000ED00;// read Cortex part number
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 47. Read UID and PID

```
/* get AT32 MCU's UID/PID base address */
#define DEVICE_ID_ADDR1 0x1FFFF7F3          //define Artery 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};

/* AT32F423 MCU type table */
const uint64_t AT32_MCU_ID_TABLE[] =
{
       0x00000012700A3240,   //AT32F423VCT7      256KB     LQFP100
       0x00000012700A3240,   //AT32F423VBT7      128KB     LQFP100
       …
};

  /* 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);

/* 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: 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 more information, please refer to the "DEBUG" section of the corresponding reference manual and *AN0016_Recognize_AT32_MCU* (path: ARTERY official website→SUPPORT→AP Note→AN0016).

2 FAQ for 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 program uses SRAM exceeding its maximum capacity threshold.
- System clock is set out of spec.

2.2 J-Link unable to recognize IC in Keil project

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

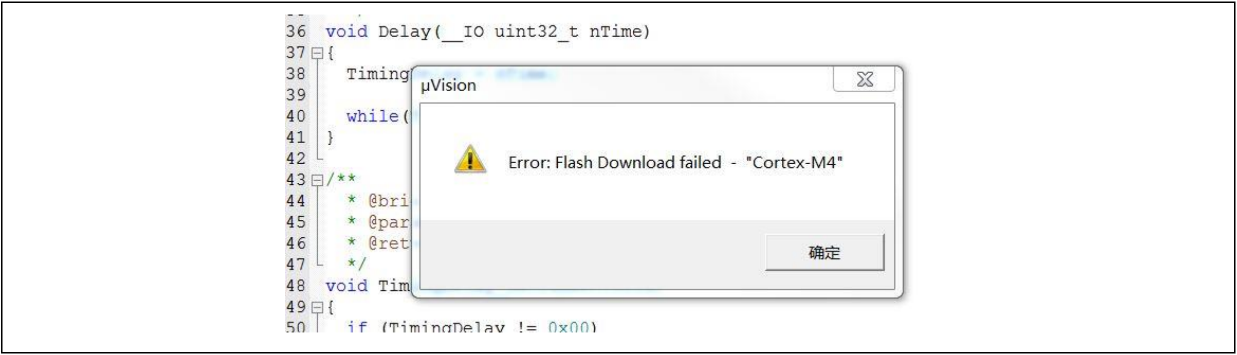
- "FAQ0008_J-Link_cannot_ find_IC", which is stored at ARTERY official website →SUPPORT→FAQ→FAQ0008.
- "FAQ0132_How_to_add_Artery_MCU_into_JLINK", which is stored at ARTERY official website →SUPPORT→FAQ→FAQ0132.

2.3 Possible problems occurred during download

2.3.1 Error: Flash Download failed–"Cortex-M4"

When Keil debugging or downloading, a warning message below pops up:

Figure 48. Flash Download failed–"Cortex- M4" during download



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

  while(
}

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

µVision

Error: Flash Download failed - "Cortex-M4"

There are several possible factors behind this pop-up message:

- Access protection is enabled. If so, unlock access protection before download.
- You may not load Flash algorithm file or choose an incorrect one. If so, add a correct Flash algorithm to Flash Download location.
- Wrong BOOT0 setting. Note that the BOOT0 must be set to 0 to boot MCU from main Flash memory.
- Older version of J-Link driver. Note that 6.20C or above is recommended for J-Link.
- JTAG/SWD pin is disabled. Please refer to Section 2.3.5 for how to resume download.

2.3.2 No Debug Unit Device found

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

2.3.3 RDDI-DAP Error

- Compiler offers a higher level of optimization. For instance, the optimization level for Keil AC6 compiler is the default “-Oz”, so it should be changed to “-O0/-O1”.
- JTAG/SWD pin is disabled. Please refer to Section 2.3.5 for how to resume download.

2.3.4 ISP serial interface gets stuck during download

When using ISP interface for download, it is stuck at a certain location so that it cannot be released.
Proceed with this issue as follows:

- Check if power supply you are using is stable or not.
- Turn to a good-quality USB-to-serial interface tool such as CH340 chip.

2.3.5 How to resume program download

After executing the following operations, users may not be able to download programs:

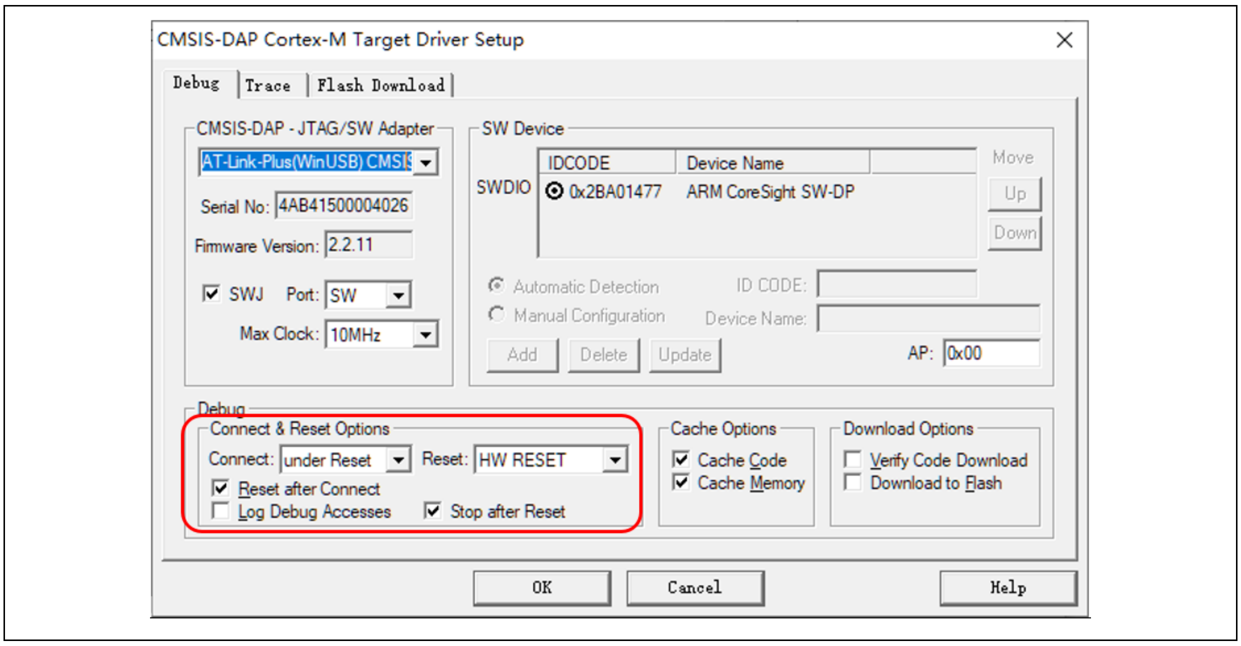
- Disable JTAG/SWD PIN disable, so that program download failed and JTAG/SWD device cannot be located.
- Enter Standby mode, so that program download failed and JTAG/SWD device cannot be located.

The principle of solution is to halt the chip when the program is not running yet.

Recommended solutions:

1. Set BOOT mode: Configure to boot from boot memory or SRAM, then reset the MCU by setting the reset pin, and then perform erase operation to resume download.
2. Use ICP tool to add AT-Link: Connect AT-Link RST pin to the MCU reset pin, then choose AT-Link connection on ICP Programmer interface, and then erase the MCU to resume download.
3. Use Keil to add AT-Link: Connect AT-Link RST pin to the MCU reset pin, then configure debug settings in Keil (as marked with red rectangle in Figure 49), and then perform erase operation to resume download.

Figure 49. Configure debug settings


CMSIS-DAP Cortex-M Target Driver Setup

Debug | Trace | Flash Download

CMSIS-DAP - JTAG/SW Adapter: AT-Link-Plus(WinUSB) CMSIS
Serial No: 4AB41500004026
Firmware Version: 2.2.11
☑ SWJ Port: SW
Max Clock: 10MHz

SW Device

	IDCODE	Device Name
SWDIO	0x2BA01477	ARM CoreSight SW-DP

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

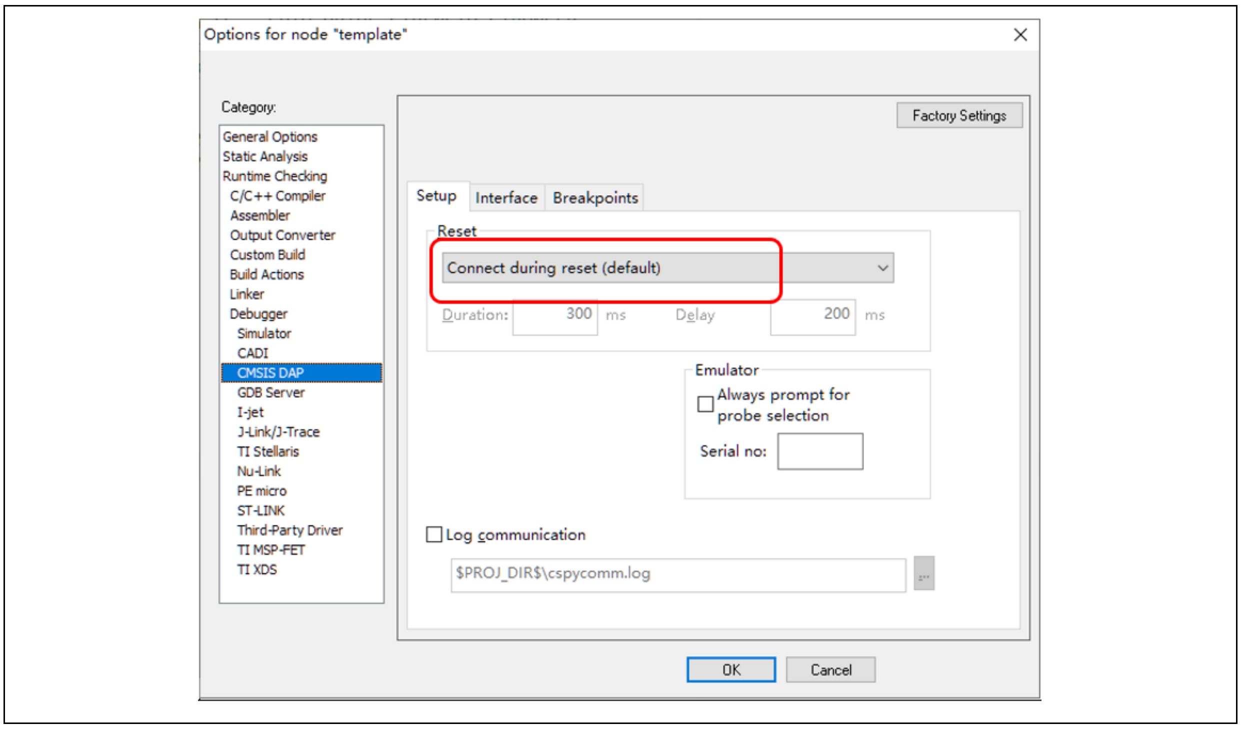
Debug
Connect & Reset Options
Connect: under Reset Reset: HW RESET
☑ Reset after Connect
☐ Log Debug Accesses ☑ Stop after Reset

Cache Options
☑ Cache Code
☑ Cache Memory

Download Options
☐ Verify Code Download
☐ Download to Flash

4. Use IAR to add AT-Link: Connect AT-Link RST pin to the MCU reset pin, then configure “CMSIS DAP” settings in IAR (as marked with red rectangle in Figure 50), and then perform erase operation to resume download.

Figure 50. Configure CMSIS DAP settings


Options for node "template"

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, J-Link/J-Trace, TI Stellaris, Nu-Link, PE micro, ST-LINK, Third-Party Driver, TI MSP-FET, TI XDS

Setup | Interface | Breakpoints

Reset: Connect during reset (default)
Duration: 300 ms Delay 200 ms

Emulator
☐ Always prompt for probe selection
Serial no:

☐ Log communication
$PROJ_DIR$\cspycomm.log

3 Security library (sLib)

3.1 Introduction

At present, as an increasing number of microcontrollers (known as MCU) require complex algorithms and middleware solutions, how to protect core algorithms and other IP codes of solution providers has emerged as one of the most important concerns in the field of MCU applications.

In response to this demand, AT32F423 series is equipped with a security library, known as sLib, with the aim of preventing important IP codes from being altered or read by end user program, so as to safeguard the rights of solution providers.

3.2 Application principles

- Any part of Flash memory can be designated as a security library (sLib) with password. This sLib is used for storing critical algorithms by solution providers while the remaining memory area can be used for secondary development by end users.
- The security library is divided into a read-only area (SLIB_READ_ONLY) and an instruction area (SLIB_INSTRUCTION). Part of or the entire sLib can be set as read-only area or instruction area.
- The SLIB_READ_ONLY area can be read through I-Code and D-Code, but it is write-protected.
- The codes in the SLIB_INSTRUCTION area can only be fetched (only executable) by MCU through I-Code. They cannot be read by reading access (including ISP/ICP/debug mode or boot from internal RAM) via D-Code, for accessing SLIB_INSTRUCTION by reading operation will return all 0xFF.
- Codes and data within sLib cannot be erased until a correct password is entered. Performing write or erase operation in case of wrong password entry will trigger a warning from EPPERR=1 of the FLASH_STS register.
- Mass erase to the main Flash memory by end users will not affect codes and data in the sLib, meaning that programs and data in this secure area will not be erased.
- After sLib feature is enabled, users can also unlock this protection through wring a correct password in the SLIB_PWD_CLR register. Once sLib is unlocked, MCU will erase the whole main memory, including sLib. This kind of design is to protect program codes against leakage even if the password set by solution providers is leaked.

3.3 How to use sLib

For details on sLib, please refer to *AN0164_AT32F423_Security_Library_Application_Note*, which is stored at ARTERY official website→SUPPORT→AP Note→AN0164.

4 Revision history

Table 1. Document revision history

Date	Version	Revision note
2023.03.20	2.0.0	Initial release.
2024.01.05	2.0.1	Updated Section 2.3.5 “How to resume program download” and added section 1.1.4 “AT32 Work Bench”.