AN0022

Getting started guide for AT32F415 series

Introduction

This user manual provides information on how to use AT32F415 MCU for project development in a quickly manner.

Applicable products：

Part number	AT32F415xx

Contents

List of tables

List of figures

1 Development environment

MCU resources download address:

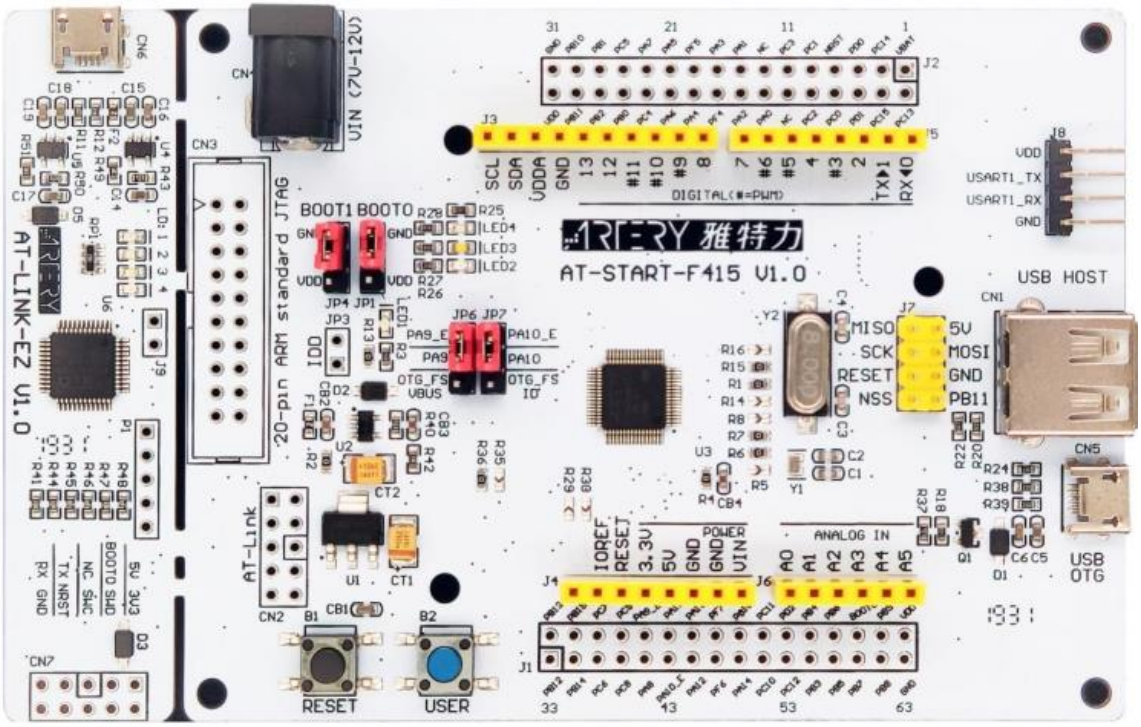
■ Visit Artery website: http://www.arterytek.com

1.1 Set up AT32 development environment

1.1.1 Debug tools

At present, AT32F415 evaluation board is equipped with AT-Link/J-Link. The following picture is for AT-START-F415 (right) with AT-Link-EZ (left).

Figure 1. AT-Link



1.1.2 Programming tools and software

AT programming tools and software: AT_Link, ICP/ISP.

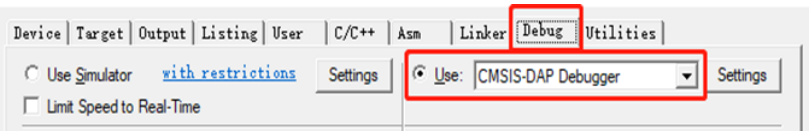
Third-party programming tools:

- Xuanwei: https://xuanweikeji.taobao.com
- Maxwiz: www.maxwiz.com.cn
- ZLG: http://tools.zlg.cn/tools
- Amo:http://www.amomcu.cn

1.1.3 AT32 KEIL and IAR development environment

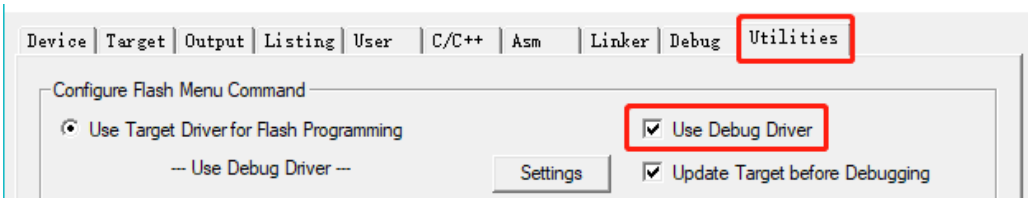
① For Keil compiling system, it is recommended to use Keil 4.74, 5.13 or above.
When using AT-Link in Keil environment, select "*CMSIS-DAP debugger* "in "*Debug*".

Figure 2. Keil Debug option



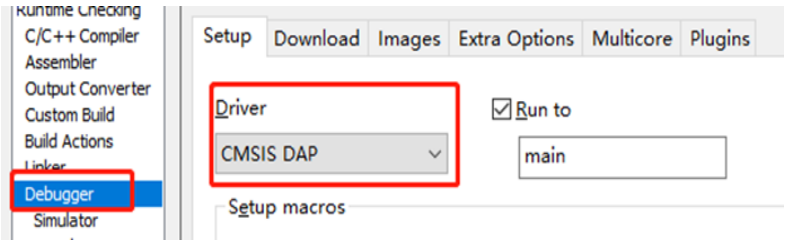
In "*Utilities*", first untick the box "*Use Debug Driver*" and then tick it again.

Figure 3. Keil Utilities option



② For IAR compiling system, it is recommended to use IAR 7.0, IAR 6.1 or above.
When using AT-Link-EZ in IAR environment, select "*CMSIS-DAP*" in "*Debugger*".

Figure 4. Select CMSIS_DAP in the IAR_Debugger



③ BSP and PACK

For detailed operation on BSP and PACK, please refer to *BSP and Pack application note* available in the BSP folder.

2 Enhanced functions of AT32F415

2.1 Prefetch buffer

Thanks to the prefetch buffer, faster CPU execution is possible as the CPU fetches one word at a time with the next word readily available in the prefetch buffer. The prefetch controller decides whether to access the Flash memory according to the available space in the prefetch buffer. The Prefetch controller starts a read operation when there is at least one free space.

Different latency should be set for different SYSCLK by setting the bit2~0 in the FLASH_ACR register.

LATENCY: Latency

These bits represent the ratio of the SYSCLK period to the Flash access time.

000: Zero wait state, if 0 <SYSCLK≤ 32 MHz

001: One wait state, if 32 <SYSCLK≤ 64 MHz

010: TWO wait states, if 64 <SYSCLK≤ 96 MHz

011: Three wait states, if 96 <SYSCLK≤ 128 MHz

100: Four wait states, if 128 <SYSCLK≤ 150 MHz

The following bond font is added in the system frequency configuration function of AT. For others, find the same location to do the same settings.

```
static void SetSysClockTo72M(void)
{
  …
  …
if (HSEStatus == (uint32_t)0x01)
{
  #if defined (AT32F415xx)
  /* Enable Prefetch Buffer */
 FLASH->ACR |= FLASH_ACR_PRFTBE;                    //Enable the prefetch buffer
 /* Flash 1 wait state */
  FLASH->ACR &= (uint32_t)((uint32_t)~FLASH_ACR_LATENCY);
 FLASH->ACR |= (uint32_t)FLASH_ACR_LATENCY_2;       //Two wait states
#endif
…
…
}
…
…
}
```

Note: The prefetch buffer must be kept on when using a prescaler different from 1 on the AHB clock.

2.2 PLL clock settings

There are two ways to configure the PLL of AT32F415 series: RCC_CFG and RCC_PLL.
Use the RCC_PLL configuration to configure more PLL clock frequency with the formula:
PLL output clock= PLL reference input clock x PLL_NS / (PLL_MS x PLL_FR).

PLL reference input clock: set to the same parameters as the actual HSE or HSI by PLL_FREF
PLL_NS: 31~500
PLL_MS: 1~15
PLL_FR: set by PLL_FR

The following bond font is added in the AT library system frequency configuration function, such as void SetSysClockTo150M (void), HSE=8 MHz.

```
static void SetSysClockTo150M(void)
{
  if (HSEStatus == (uint32_t)0x01)
  {
    ...
    ...
    /*    PLL configuration: PLLCLK = (HSE * 75) / (1 * 4) = 150 MHz */
    RCC->CFG &= RCC_CFG_PLLCFG_MASK;
    RCC->CFG |= (uint32_t)(RCC_CFG_PLLRC_HSE);          //HSE clock used as PLL input clock
    RCC_PLLconfig2(PLL_FREF_8M, 75, 1, PLL_FR_4);       //set PLL output clock
      ...
      ...
}
```

Note: When using RCC_PLL, 500MHz <= PLL reference input clock x PLL_NS/PLL_MS <= 1000MHz must be met.

2.3 PLL auto step-by-step frequency switching function

When the PLL embedded in the AT32F415 series is greater than 108 MHz, the auto step-by-step frequency switching function should be operated. The following bond font is added in the system frequency configuration function of AT library, such as void SetSysClockTo150M (void). For others, find the same location to do the same settings.

```
static void SetSysClockTo150M(void)
{
  …
  …
  if (HSEStatus == (uint32_t)0x01)
  {
…
…
      /* Wait till PLL is ready */
      while((RCC->CTRL & RCC_CTRL_PLLSTBL) == 0)
      {
      }
#if defined (AT32F413xx) || defined (AT32F415xx)
      RCC_StepModeCmd(ENABLE);        // enable the auto step-by-step frequency switching function
#endif
      /* Select PLL as system clock source */
      RCC->CFG &= (uint32_t)((uint32_t)~(RCC_CFG_SYSCLKSEL));
      RCC->CFG |= (uint32_t)RCC_CFG_SYSCLKSEL_PLL;
      /* Wait till PLL is used as system clock source */
      while ((RCC->CFG & (uint32_t)RCC_CFG_SYSCLKSTS) != RCC_CFG_SYSCLKSTS_PLL)
      {
}
#if defined (AT32F413xx) || defined (AT32F415xx)
      RCC_StepModeCmd(DISABLE);     //disable the auto step-by-step frequency switching function
#endif
  }
  …
}
```

Note: If the auto step-by-step frequency switching feature is enabled, it must be disabled after the clock is switched, and enable and disable must be paired.

2.4 Encryption (Read protection)

2.4.1 Read protection

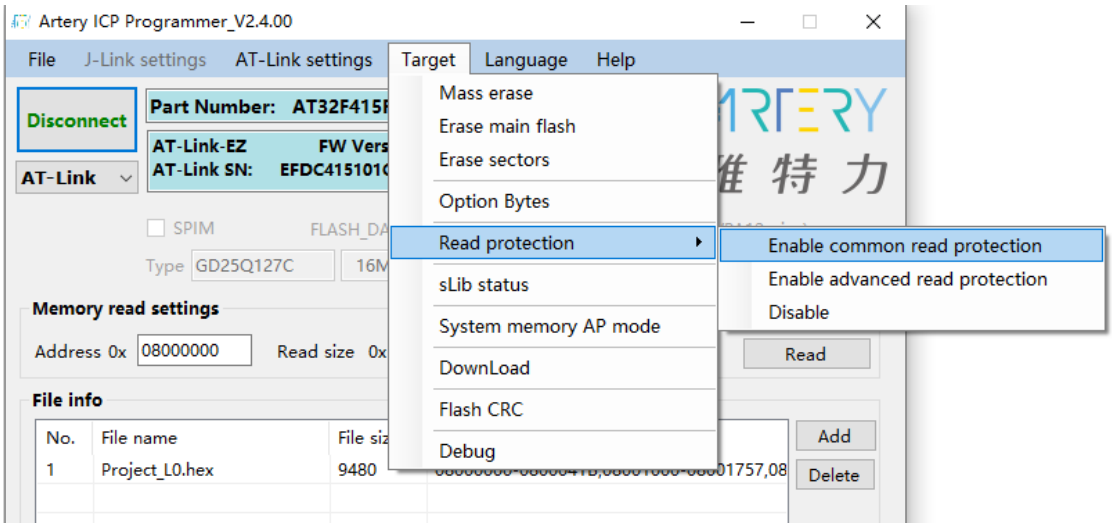
Read protection, commonly referred to as “encryption”, acts on the entire Flash storage area. Once the read protection is set in the Flash, the embedded Flash storage area can only be read through the normal execution of the program instead of JTAG or SWD. When the read protection is disabled using ISP/ICP tool, the chip will erase the Flash.

ISP/ICP tool can be used to enable/disable read protection as follows:

- ICP tool

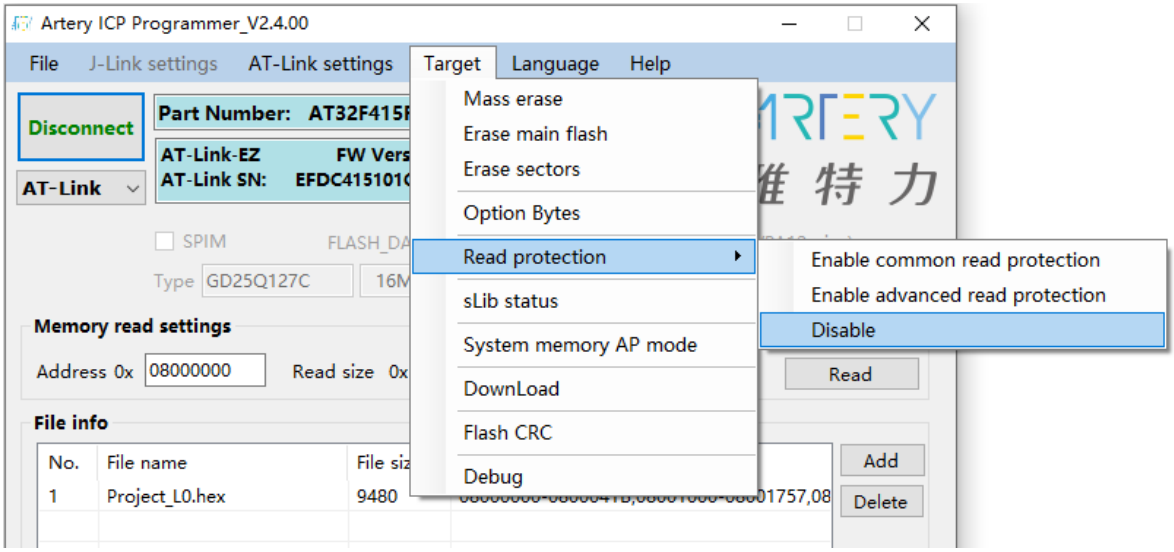
Read protection: “***Target***”---“***read protection***”---“***enable protection***”

Figure 5. ICP enable read protection



Disable read protection: “***Target”***---“***read protection***”---“***disable read protection***”

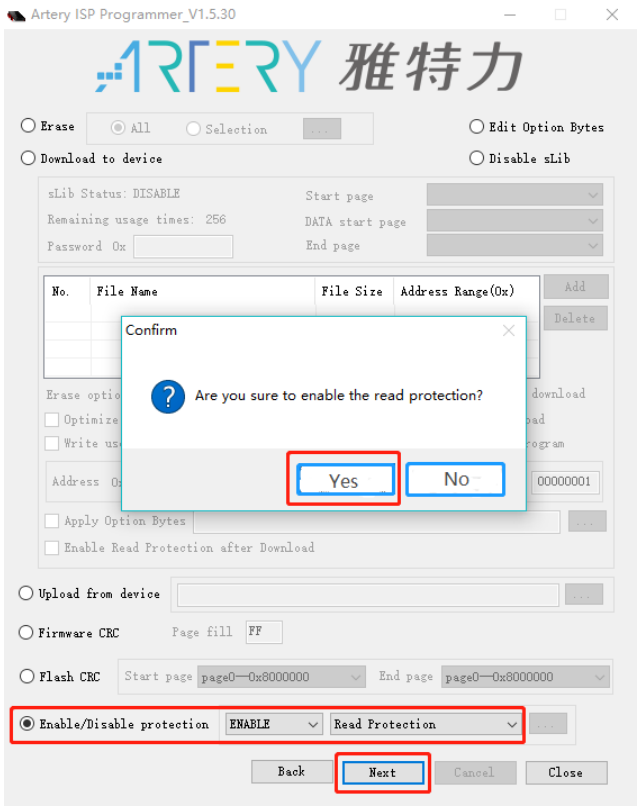
Figure 6. ICP disable read protection



■ Artery ISP Programmer tool

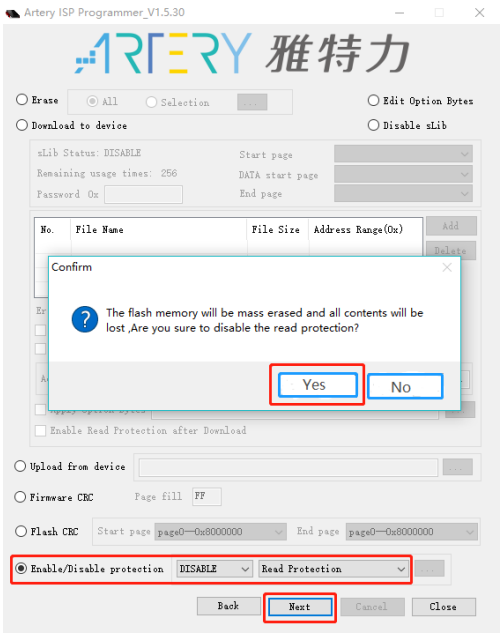
Read protection: “*enable/disable protection*”---“*enable---read protection*”---“*Next*”---“*Yes*”, and the program is encrypted.

Figure 7. ISP enable read protection



Disable read protection: “*enable/disable protection*”---“*disable-read protection*”---“*Next*”—“*Yes*”, and then the Flash can be unencrypted.

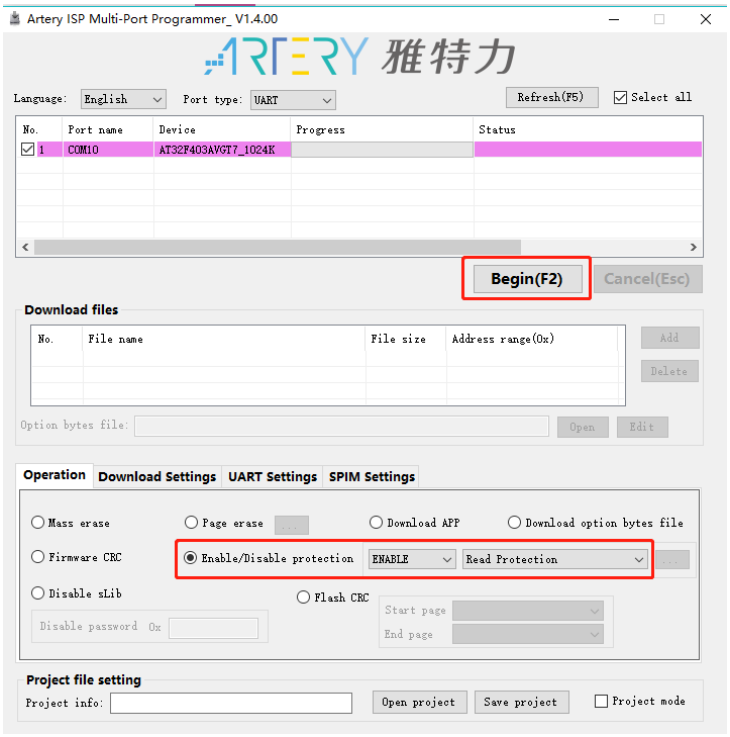
Figure 8. ISP disable read protection



■ Artery ISP Multi-Port Programmer

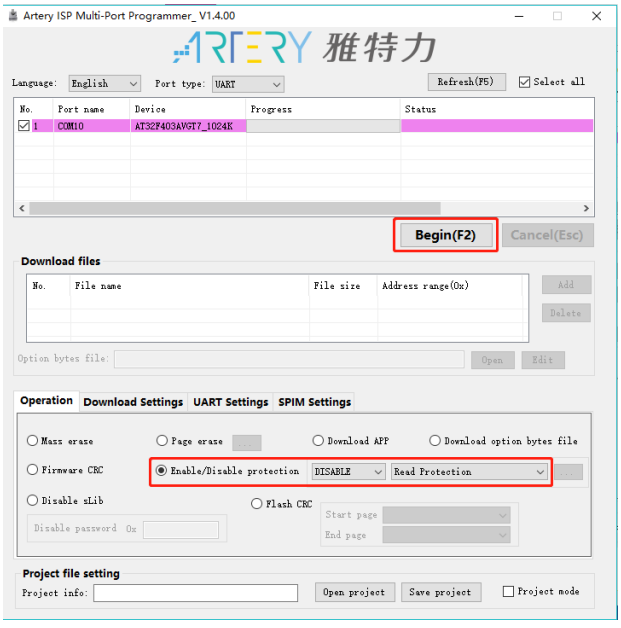
Read protection: "*enable/disable protection*"---"*enable-read protection*"---click on "Begin", and then the program is encrypted.

Figure 9. ISP Multi-Port Programmer enable read protection



Disable read protection: "*enable/disable protection*"--- "*disable-read protection*"—click on "*Begin*", then the Flash is unencrypted. The read protection cannot be disabled by the erase operation.

Figure 10. ISP Multi-Port Programmer disable read protection



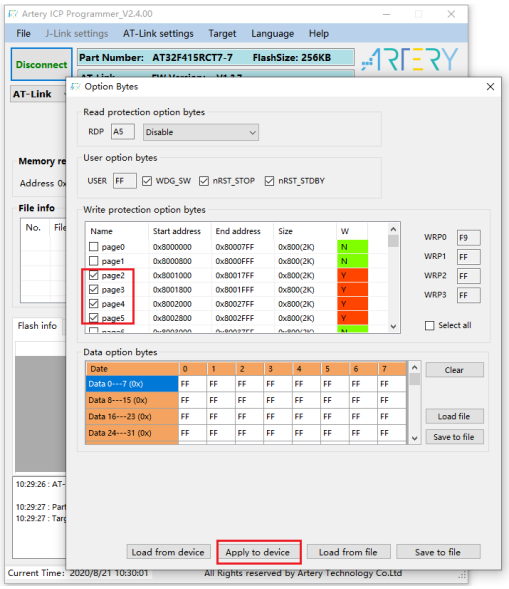
2.4.2 Write protection

Write protection acts on the whole Flash storage area or some pages. Once the write protection is set in the Flash, the embedded Flash storage area cannot be written in any way. ISP/ICP tools can be used to enable/disable the write protection.

① Artery ICP Programmer tool

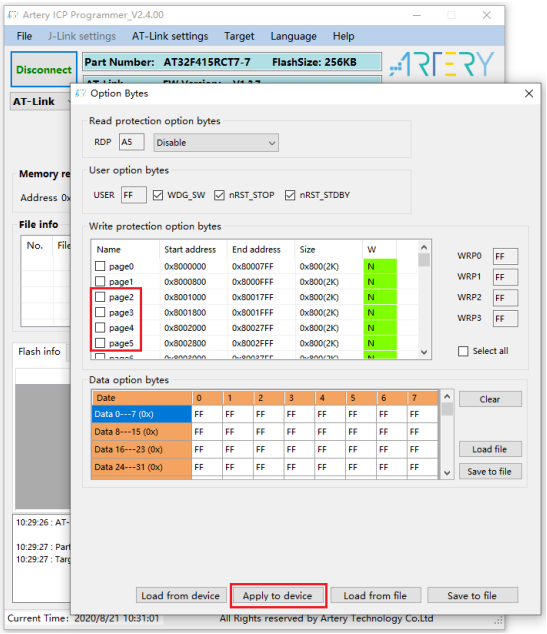
Enable write protection: click "*Target*"---"*Write protection option byte*"----Select the pages to be protected---"*Apply to device*"

Figure 11. ICP pgrammer enable read protection



Disable write protection: click "*Target*"---"*Write protection option byte*"----Select the pages to be canceled ---"*Apply to device*". The write protection cannot be disabled by the erase operation.

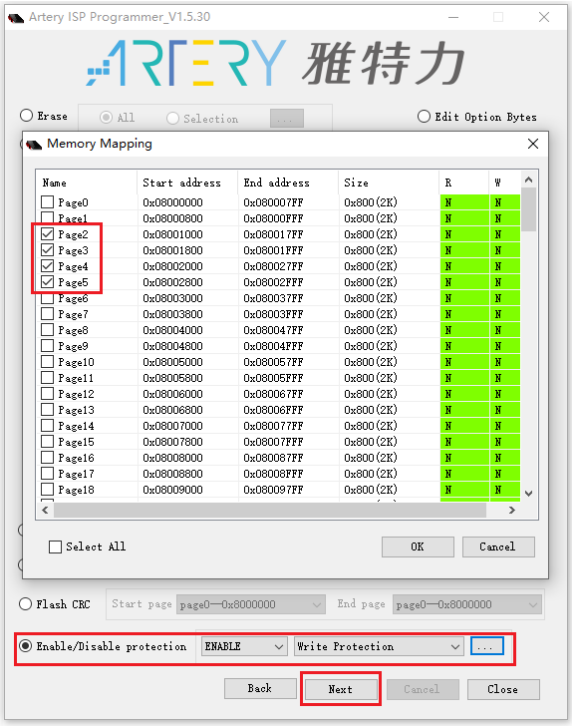
Figure 12. ICP pgrammer disable read protection



② Artery ISP Programmer tool

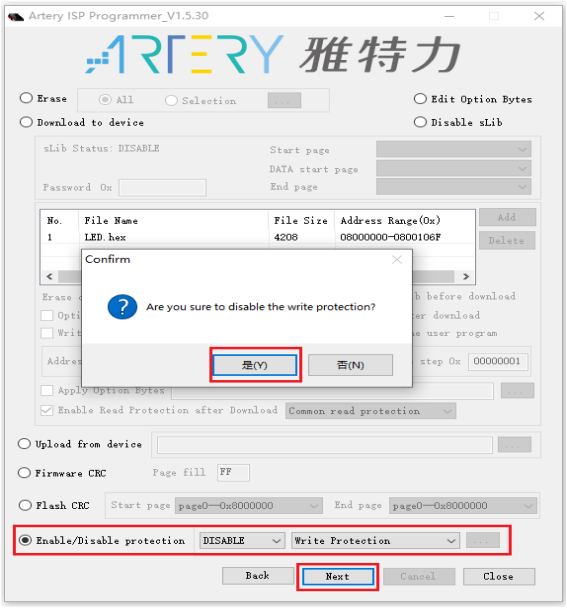
Enable write protection: check "*Enable/Disable protection*"---Select "*Enable*"--- "*Write protection*"----Select the pages to be protected---Click "*OK*"---Click *"Next*"---"*Yes*".

Figure 13. ISP pgrammer enable read protection



Disable write protection: check "*Enable/Disable protection*"---select "*Disable*"--- "*Write protection*"---Click "*Next*"---"*Yes"*. The write protection cannot be disabled by the erase operation.

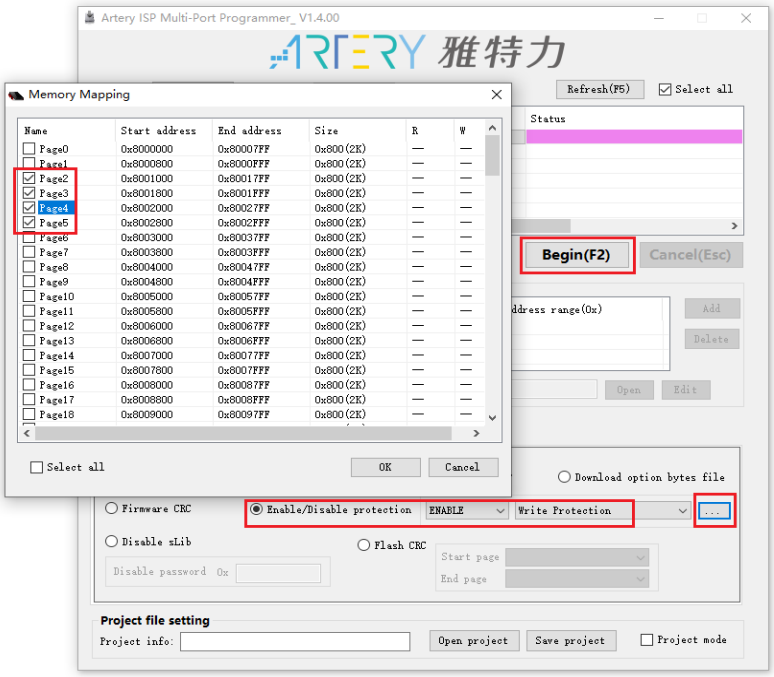
Figure 14. ISP programmer disable read protection



③ Artery ISP Multi-Port Programmer tool

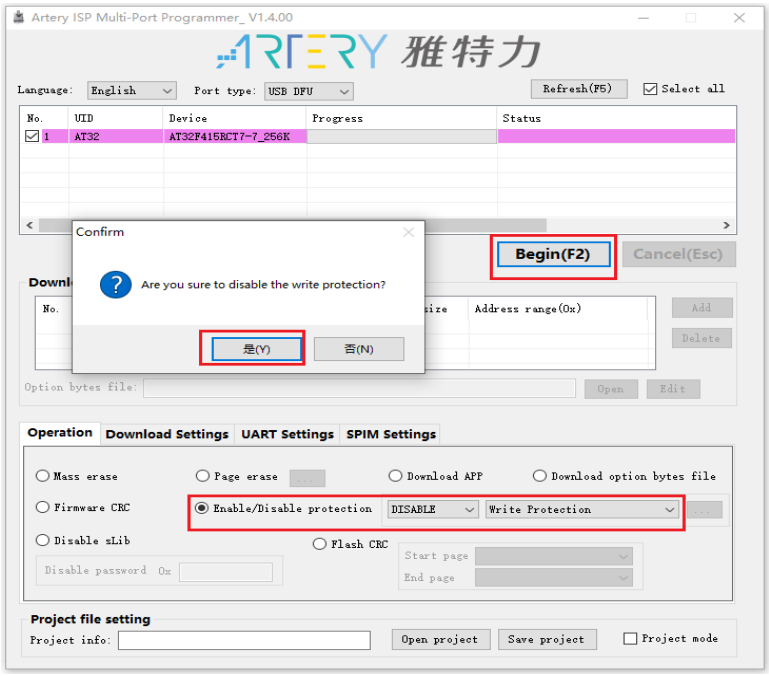
Enable write protection: check "*Enable/Disable protection*"---select "*Enable*"---- "*Write protection*"----select the pages to be protected---click "*OK*"---click "*Begin*"---click "*Yes*".

Figure 15. ISP Multi-Port programmer enable read protection



Disable write protection: check "*Enable/Disable protection*"---select "*Disable*"--- "*Write protection*"---Click "*Begin*"---*Yes*. The write protection cannot be disabled by the erase operation.

Figure 16. ISP Multi-Port programmer disable read protection



2.4.3 Security library (sLib)

- **Overview**

At present, more and more microcontroller applications need to use complex algorithms and middleware solutions, therefore, how to protect the core algorithms and intellectual property codes (IP-Code) developed by software solutions providers has become a very important subject in the microcontroller applications.

In response to this important demand, AT32F415 series provide the security library function (sLib) to prevent important IP-Code from being modified or read by end-user programs to achieve protection.

- **Application principle**

- Supports the use of password to protect the specified program area (security library) in the main Flash. The IDH can store the core algorithm into this security library to implement protection, with the remaining blank area for end customers to conduct secondary development.
- The security library is divided into instruction security library (sLib_Code) and data security library (sLib_Data), and part or the whole security library can be selected to store instructions, but the whole security library is not supported to store data.
- The program code in the sLib_Code can only be fetched by the MCU through the I-Code bus (can only be executed), and cannot be read through the D-Code bus (including ISP/ICP debug mode and programs started from internal RAM). All the values read are 0xFF when accessing to sLib_Code by reading data.
- The data in the sLib_Data can only be read through D-Code, but not written.
- The program code and data in the security data cannot be erase until a correct password is entered. If attempt to write to or erase the security library in the event of a wrong password, the WRPRTFLR bit of the FLASH_STS register will issue a warning by setting “1”.
- The program code and data in the security library would not be erased when the end user performs the whole chip erase on the main Flash memory.
- When the protection function of security library is activated, it can be disabled by writing the previously set password in the sLib_PSW register. When the protection function of security library is disabled, the erase operation will be performed on the whole Flash memory (including the contents in the security library). Therefore, even if the password set by IDH is leaked, the program code will not be disclosed.

- **How to operate security library**

Please refer to AT32F415 sLib Application note for further details.

2.5 Set the system memory as an extended main memory

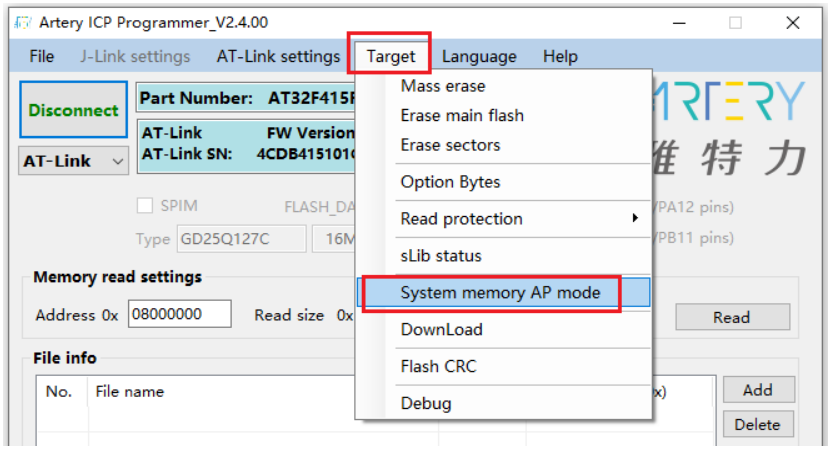
By default, the system memory is used as BOOT mode to store the original factory-cured startup code. However, new functions have been added in the AT32F415 products, where the system memory can also be selected as the extended area (AP mode) of the main memory to store user-defined codes.

Note: the system memory AP mode can only be set once and is irreversible. After setting, the original system memory BOOT mode cannot be restored.

During product development, Artery ICP Programmer is used to set the system memory as an extended main memory, with the following procedures:

- Connect J-Link or AT-Link simulator to AT-START-F415 board and power on.
- Turn on the ICP programmer, and select J-Link or AT-Link to connect.
- In menu bar: click “Device operation”---select “system memory AP mode”.

Figure 17. System memory AP mode



- To prevent incorrect operation, you need to manually enter the encryption key 0xA35F6D24. Then, a success or failure message will appear in the “Flash info” window.

Figure 18. Flash information window

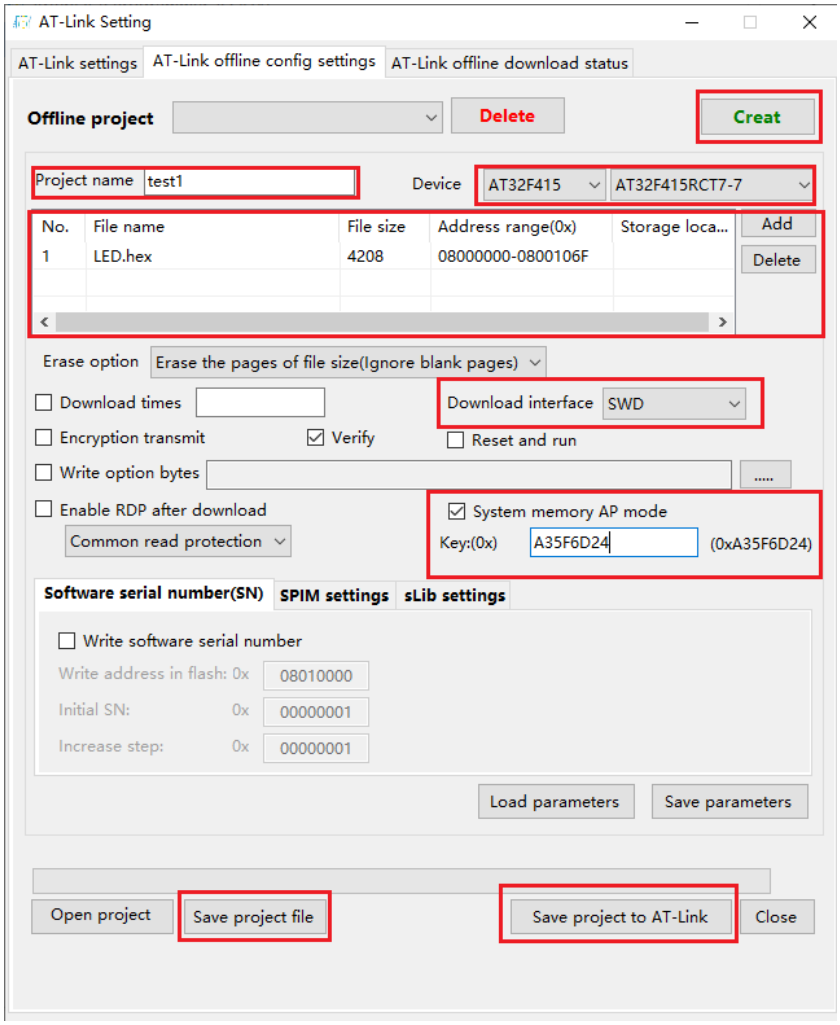
In mass production, the Artery ICP Programmer is used to set the system memory as an extended main memory as follows:

- Connect AT-Link (the AT-Link EZ in the picture above does not support offline programming, so choose non-EZ version of AT-Link) simulator to the AT-START-F415 board and power on.
- Activate the ICP programmer, and select AT-Link to connect.
- In menu bar: AT-Link settings----click “Offline project configuration” to generate offline projects.
- The following are the steps that generate offline project:

1. Click “create a new project”
2. Enter the project name in the “Project name”
3. Select the MCU part number in the “Supported MCU”
4. Add .hex files
5. Select SWD in the “Download communication interface”
6. Check the “System memory AP mode” and enter the encryption key
7. Click the “Save project files or “Save the project to AT-Link”

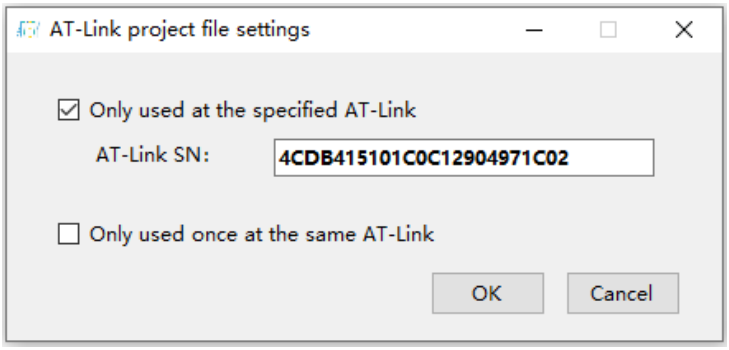
Other options are set depending on the actual needs.

Figure 19. How to generate offline project



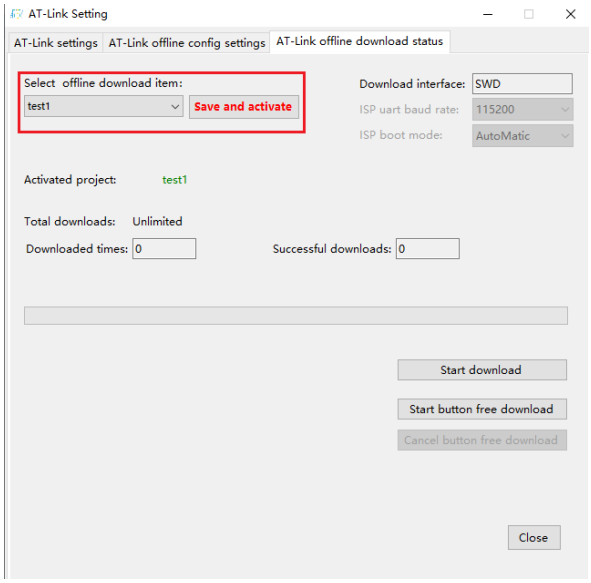
● If you click the "*Save the project file*" in step 7, the project will be saved as a file in the format of .actp, which is convenient for loading to other AT-Link. A window as shown below will pop up during the operation. If you tick the "*Only used in the specified AT-Link*", this project file is bound to AT-Link and can only be used in the bound AT-Link, and the AT-Link serial number should be set; if you tick the "*Only used once at the same AT-Link*", it means that this project file can only be used once in the same AT-Link.

Figure 20. How to save offline project



● In step 7, if the operation "*Save the project to AT-Link*" is successful, in the offline download status monitoring window, select the project name in the "*Select offline download item*", and click "*Save and activate*", you can start programming.

Figure 21. Activate offline project



For demo on using the user program in the system memory, please refer to BSP under the location:

AT32F4xx_StdPeriph_Lib_V1.x.x\Utilities\AT32F415_SysMem_AP_Demo.

For ICP user manual, please refer to:

ICP_Programming_Tool_Vx.x.x\Document\UM_ICP_Programmer.

For AT-Link user manual, please refer to:

AT-Link_Vx.x.x\User Manual\ AT-Link_User_Manual_SC.

3 How to distinguish AT and other ICs

① Read the Cortex-M series CPU ID to distinguish M0, M3 and M4 core.

```
    i = *(uint32_t *)0xE000ED00;//read PID
  if((i & 0xc241) == 0xc241)
      printf("This chip is Cortex-M4.\r\n");
  else
       printf("This chip is Other Device.\r\n");
```

② Read PID and UID to distinguish

```
/* Obtain the base address of AT32 MCU PID/UID*/
  #define DEVICE_ID_ADDR1 0x1FFFF7F3
  #define DEVICE_ID_ADDR2 0xE0042000
/* To store ID */
    uint8_t     ID[5] = {0};

/* AT32 mcu type table */
const uint64_t AT32_MCU_ID_TABLE[] =              /* 415 mcu */
{
    0x0000000570030240,   //AT32F415RCT7      256KB     LQFP64
    0x00000005700301C1,   //AT32F415RBT7      128KB     LQFP64
    …
 };
 /* Get PID/UID */
ID[0] = *(int*)DEVICE_ID_ADDR1;
ID[1] = *(int*)(DEVICE_ID_ADDR2+3);
ID[2] = *(int*)(DEVICE_ID_ADDR2+2);
ID[3] = *(int*)(DEVICE_ID_ADDR2+1);
ID[4] = *(int*)(DEVICE_ID_ADDR2+0);
/*  组合 PID/UID */
 AT_device_id =
  ((uint64_t)ID[0]<<32)|((uint64_t)ID[1]<<24)|((uint64_t)ID[2]<<16)|((uint64_t)ID[3]<<8)|((uint64_t)ID[4]<<0)
  ;

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

4 FAQs during download and compiling

4.1 Error occurred during the downloading

4.1.1 Error: Flash Download failed – “Cortex-M4”

An error pops up during KEIL emulation or download:

Figure 22. Flash Download failed – “Cortex- 4”

Several possible reasons are as follows:

A. The read protection is enabled: you need to disable MCU read protection before download;

B. Select a wrong Flash file algorithm: you need to add a correct Flash file algorithm at Flash Down location.

C. Select wrong Boot0 and Boot1: Boot0 and Boot1 pin level should be set to Boot0=0 and Boot1=0 respectively so that MCU starts from the main Flash memory;

D. J-Link driver version is too old: it is recommended to use a version above 6.20C;

E. JTAG/SWD PIN is disabled, please refer to the section “4.1.3 AT32 resume download.

4.1.2 ISP serial interface gets stuck during download

When the ISP serial interface is used to download, it occasionally gets stuck, causing the PC not to release the serial port.

Solutions:

- Power supply is not stable;
- Use a better USB-to-serial interface tool, such as CH340 chip.

4.1.3 AT32 resume download

When using AT32F415, users may not be able to download the program after the following operations:

■ After the JTAG/SWD PIN is disabled, the program cannot be downloaded and the JTAG/SWD device cannot be found.

■ After entering Standby mode, the program cannot be downloaded and JTAG/SWD device cannot be found.

Here we provide the solutions in KEIL and IAR environment:

Solution 1: switch boot mode

Switch the boot mode to Boot[1:0]=01b to boot in the system memory, and use ISP tool to resume download.

Solution 2: ICP tool and AT-Link-EZ

AT-Link-EZ is specially designed for AT32, so ICP and AT-Link-EZ can use resume download.

5 Revision history

Table 1. Document revision history

Date	Revision	Changes
2019.08.23	1.00	Initial release
2019.11.22	1..01	Added 2.5 section for system memory description and updated AT32 resume download.

IMPORTANT NOTICE – PLEASE READ CAREFULLY

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

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

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

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