**AN0012**
Application Note

# AT32 USB MSD IAP

## Introduction

USB MSD IAP is a firmware upgrading tool that is independent of other hosts and can be directly connected to PC or mobile phone to upgrade the firmware in the device. This application note describes the principle of USB MSD IAP and how to implement it.

*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 | AT32 MCU with USB |

# Contents

## List of Tables

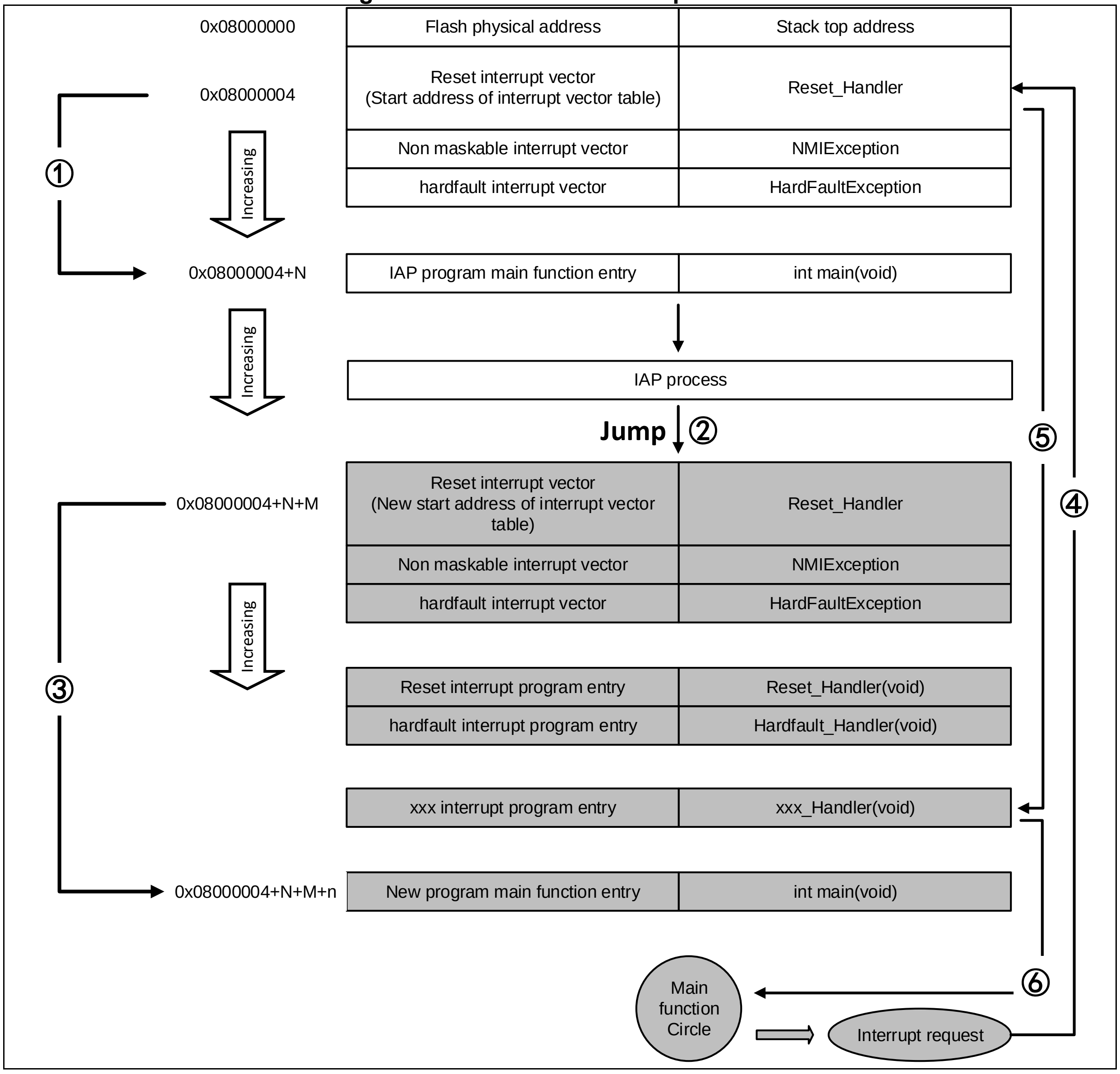
# List of Figures

# 1 Principle of IAP upgrade

IAP refers to in-application programming that makes it possible for the user application to program the user Flash at runtime so as to update the firmware in the product through the reserved communication interfaces. To implement IAP function, it is necessary to write two project codes when designing the firmware program. The first code receives program or data through some communication interfaces (such as USB or USART) to update the second code, without executing normal functional operations; the second is the actual functional code. Both codes are programmed in the User Flash simultaneously. After the microcontroller is powered, it is the first code to start running as follows:

1) Check if it is necessary to update the second code
2) If unnecessary, go to step 4)
3) Execute update operation
4) Jump to the second code for execution

**Figure 1. IAP code execution process**


0x08000000
Flash physical address
Stack top address
0x08000004
Reset interrupt vector (Start address of interrupt vector table)
Reset_Handler
Non maskable interrupt vector
NMIException
hardfault interrupt vector
HardFaultException
Increasing
①
0x08000004+N
IAP program main function entry
int main(void)
IAP process
Jump
②
0x08000004+N+M
Reset interrupt vector (New start address of interrupt vector table)
Reset_Handler
Non maskable interrupt vector
NMIException
hardfault interrupt vector
HardFaultException
③
Reset interrupt program entry
Reset_Handler(void)
hardfault interrupt program entry
Hardfault_Handler(void)
xxx interrupt program entry
xxx_Handler(void)
0x08000004+N+M+n
New program main function entry
int main(void)
④
⑤
⑥
Main function Circle
Interrupt request

From this flowchart, after reset, AT32 microcontroller still fetches the address of the reset interrupt vector from the 0X08000004 and jumps to the reset interrupt service program. After completing the reset interrupt service program, it jumps to the main function of IAP (shown in ①) to execute IAP (that is, write a new APP code into AT32 Flash, marked in grey. The start address of the reset interrupt vector of the new program becomes 0X08000004+N+M), and then jumps to the reset vector table of the new program to fetch its address, and jumps to run the new reset interrupt service program, and then jumps to the new main function (shown in ② and ③). Similarly, the main function is an endless loop, and it should be noted that AT32 Flash has two interrupt vector tables in different positions.

If CPU received an interrupt request in the process of executing main function, the PC pointer still forcibly jumps to the interrupt vector table of the address 0X08000004, instead of the new one (shown in ④). Then, the program jumps to the new interrupt service program corresponding to the interrupt source based on the interrupt vector table offset value configured, as shown in ⑤. After completing the interrupt service program, it returns to the main function to continue running, as shown in ⑥.

Based on the above analysis, we know that two requirements must be met for IAP:

1) The new program must start at the address with offset x following IAP program
2) The interrupt vector table of the new program must be moved accordingly on the basis of an offset x.

# 2 AT32 USB MSD IAP introduction

USB MSD IAP is a firmware upgrading tool that is independent of other hosts and can be directly connected to PC or mobile phone to upgrade the firmware in the device.
Principle: FLASH is virtualized as a storage device for PC access.
Complete very simple procedures as follows:

1) Connect the USB interface to PC
2) PC recognizes the drive letter "AT32 IAP"
3) Copy the firmware to be updated into AT32 IAP drive letter
4) End of upgrade

## 2.1 AT32 USB MSD IAP features

- At present, IAP uses 20K byte of reserved space, so the start address of APP should be after 20K;
- Use USB massive storage device for virtual device;
- After upgrade, automatically reset USB device and return to upgrade state;
- Automatically read and perform CRC after download to ensure the correctness of firmware;
- Download address can be configured (aligned per page 2K, and must be larger than the reserved IAP address);
- Support windows, linux, Android and other systems;
- Jump to APP for execution after upgrade completion;
- Support BIN file upgrade
- Support HEX file upgrade (included in the next version);
- Support encryption file upgrade (included in the next version).

## 2.2 Program design

### 2.2.1 Address space

**Table 1. Address space distribution**

| ITEM | Address |
|---|---|
| USB MSD IAP | 0x08000000 ~0x08005000 |
| APP available address | 0x08005000~…… |

### 2.2.2 Upgrade status

After being connected to the host, there will be TXT files to appear in the drive letter indicating the latest upgrade status by their respective different file name.
Ready to upgrade (Ready.TXT)
Upgrade success (Success.TXT)
Upgrade failed (Failed.TXT)
Unknown file or error (Unknown.TXT)
Upgrade file is larger than FLASH size (Large.TXT)
*Note: It must be remained in the Ready.TXT state that the device can be upgraded; otherwise, it would not.*

### 2.2.3 Upgrade BIN file name format

1. Specify the download address (format 1)
   File name format: (1Byte)A+(6Byte)offset+.BIN
   For example, if you need to download a BIN file to the address space starting from 0x08005000, the file should be named as A005000.BIN

*Note: 6Byte offset address needs to be located within the available space of APP; otherwise, the default APP start address in the IAP is used for upgrade.*

*When the format 1is not met, IAP will use internal default APP start address for upgrade.*
*For example, ABCDEFG.BIN, A11111.BIN, jkakkkddkfj.BIN.*

### 2.2.4 Upgrade HEX file format (suffix .HEX)

This section will be included in the next version.

### 2.2.5 Upgrade encryption file format (suffix .SEC)

This section will be included in the next version.

### 2.2.6 Option Byte flag indicates upgrade status

The HID[0] in the Option Byte records whether the upgrade is successful. Configure HID[0]=1 for entering IAP, and HID[0]=0 when upgrade is completed.
When the device is activated, it will automatically judge whether the firmware upgrade is finished through HID[0]; if yes, it jumps to APP address for execution; if not, keep running IAP.

### 2.2.7 Jump to APP code for execution

When the firmware is successfully downloaded to Flash, there are two methods for jumping to user code for execution.

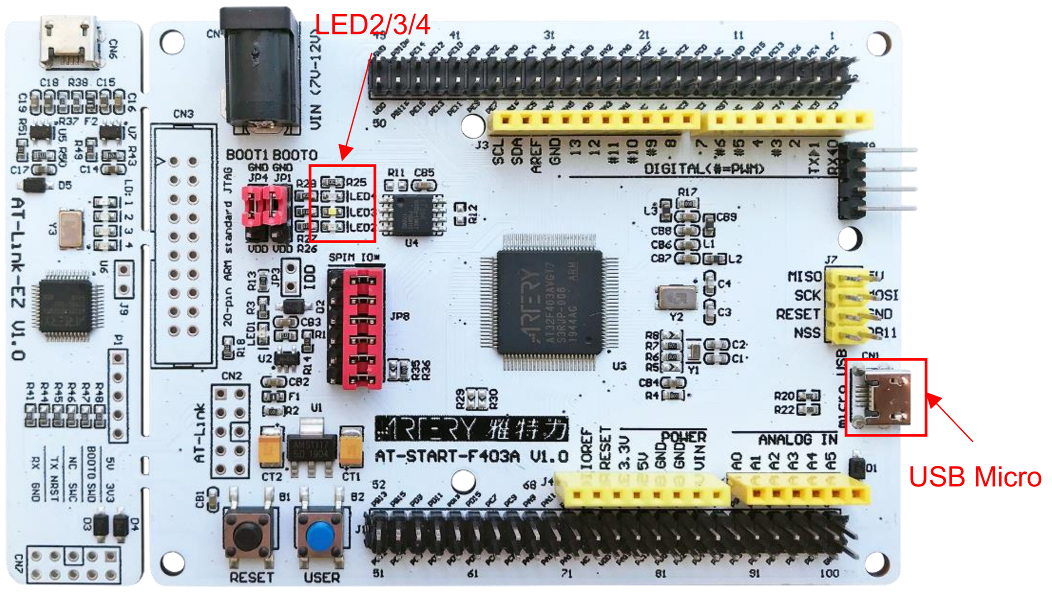
1) Reset
2) Copy a JUMP.TXT file to the identified AT IAP virtual disk. Note that the file size must not be 0.

# 3 Use AT32 USB MSD IAP for upgrade

## 3.1 Hardware resources

1) LED2/LED3/LED4
2) USB(PA11/PA12)
3) AT-START-F403A V1.0 demo board

**Figure 2. AT-START-F403A**



*Note: this IAP demo is based on the hardware conditions of AT32F403A. If the user wants to apply it to other AT32 products, please modify the corresponding configurations.*

## 3.2 Software resources

1) Source Code
   - AN0012_SourceCode_V2.0.0\utilities\AN0012_demo, IAP source code
   - AN0012_SourceCode_V2.0.0\libraries, AT32 peripheral library
   - AN0012_SourceCode_V2.0.0\middlewares, other resources
2) Doc
   - AN0012_USB_MSD_IAP_V2.x.x.docx

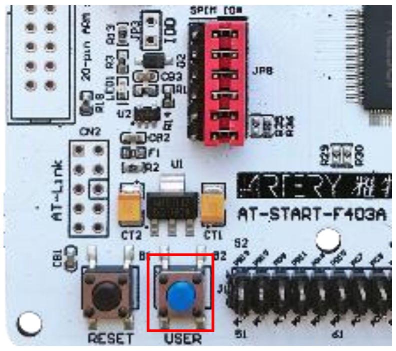
*Note: All projects are built around keil 5. If the user needs to apply it to other compiling environment, please refer to AT32F403A_407_Firmware_Library\project\at_start_f403a\templates (such as IAR6/7/8, keil 4/5) for a simple change.*

## 3.3 Operation procedures

**1) Enter IAP Mode**

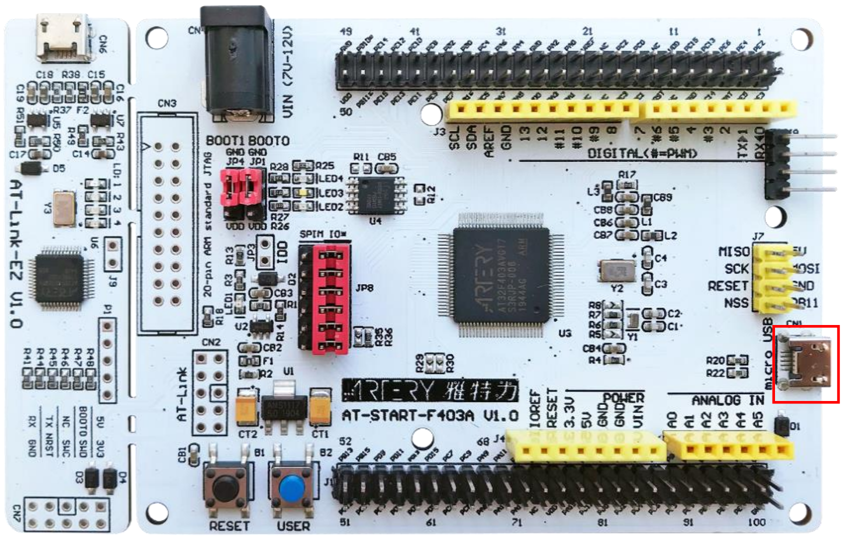
If you have upgraded the firmware, press and hold the User button and then press Reset button to enter IAP mode (LED4 blinking); otherwise, directly reset to enter IAP mode.

Figure 3. User button for entering IAP Mode



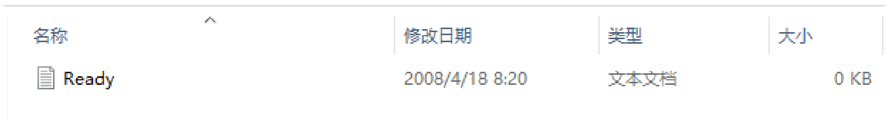
**2) Connect USB device to PC**

Figure 4. USB device is connected to PC



**3) PC identifies AT32 IAP, and Ready.TXT appears.**

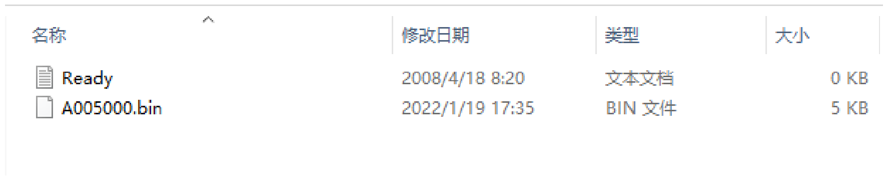
Figure 5. PC identifies AT32 IAP



| 名称 | 修改日期 | 类型 | 大小 |
|---|---|---|---|
| Ready | 2008/4/18 8:20 | 文本文档 | 0 KB |

**4) Copy a BIN file to the disk directory.**

Figure 6. Copy Bin file

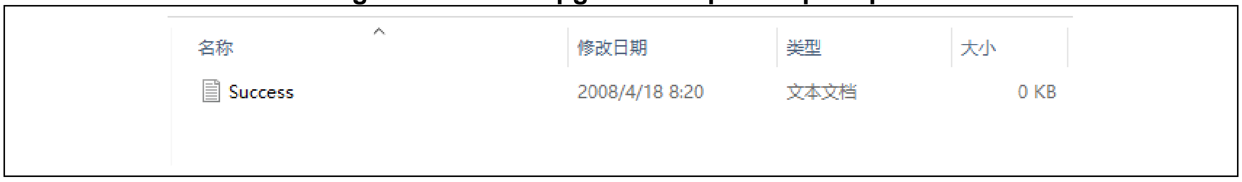


| 名称 | 修改日期 | 类型 | 大小 |
|---|---|---|---|
| Ready | 2008/4/18 8:20 | 文本文档 | 0 KB |
| A005000.bin | 2022/1/19 17:35 | BIN 文件 | 5 KB |

**5) Reset USB after the upgrade is completed.**

In this case, re-open the disk, and Success.TXT exists.

Figure 7. Device upgrade completion prompt



| 名称 | 修改日期 | 类型 | 大小 |
|---|---|---|---|
| Success | 2008/4/18 8:20 | 文本文档 | 0 KB |

**6) End of upgrade.**

In this case, reset or copy a JUMP.TXT file (size≠0) to the identified AT IAP virtual disk, then it will jump to user code.

After one upgrade, regardless of success or failure, if you need to upgrade again, it is necessary to reset the entire device to enter Ready.TXT status.

# 4 Revision history

Table 2. Document revision history

| Date | Version | Revision note |
|---|---|---|
| 2022.1.25 | 2.0.0 | Initial release |