PM0006
Programming Manual

AT32 Bootloader USB DFU Protocol

Introduction

This user manual describes the USB DFU protocol used in the AT32 MCU bootloader, detailing the supported DFU protocol commands. For more information about the USB hardware resources and requirements for your device bootloader, refer to the document PM0007_AT32_Bootloader_Program_Manual.

Contents

1 USB DFU bootloader requests 5

2 DFU bootloader commands 6

3 DFU_UPLOAD request commands 8

3.1 Get command 9

3.2 Read memory 10

3.3 Get CRC 10

3.4 Get sLib status 11

4 DFU_DNLOAD requests 12

4.1 Write Memory 14

4.2 Set Address 16

4.3 Access Protect 17

4.4 Access Unprotect 17

4.5 Erase and Program Protect command 18

4.6 Erase and Program Unprotect 19

4.7 Erase command 20

4.8 Jump command 22

4.9 Firmware CRC command 23

4.10 Enable SPIM command 24

4.11 Enable sLib 25

4.12 Disable sLib 26

4.13 Reset Device 27

4.14 Advanced Access Protect 28

5 Revision history 29

List of tables

Table 1 DFU requests 5
Table 2 DFU class-specific requests 5
Table 3 DFU bootloader commands 6
Table 4 Data received after Get command 9
Table 5 DNLOAD command conditions 12
Table 6 Erase index 20
Table 7 Document revision history 29

List of figures

Figure 1 DFU_UPLOAD request: device side 8
Figure 1 DFU_UPLOAD request: host side 8
Figure 3 DFU_DNLOAD request: device side 13
Figure 4 DFU_DNLOAD request: host side 14
Figure 5 Write Memory: device side 15
Figure 6 Set Address: device side 16
Figure 7 Access Protect: device side 17
Figure 8 Access Unprotect: device side 18
Figure 9 Erase and Program Protect: device side 19
Figure 10 Erase and Program Unprotect: device side 19
Figure 11 Erase: device side 21
Figure 12 Jump: device side 22
Figure 13 Firmware CRC: device side 23
Figure 14 Enable SPIM: device side 25
Figure 15 Enable sLib: device side 26
Figure 16 Disable sLib: device side 27
Figure 17 Reset Device: device side 28
Figure 18 Advanced Access Protect: device side 28

1 USB DFU bootloader requests

AT32 Bootloader supports the USB DFU protocol and requests. For more details concerning these requests, refer to the *Universal Serial Bus Device Upgrade Specification for Device Firmware Upgrade Version 1.1*

Table 1 DFU requests

Request	Code	Description
DFU_DETACH	0x00	Requests the device to leave DFU mode and enter the application
DFU_DNLOAD	0x01	Requests data transfer from host to the device in order to load them into device internal flash memory
DFU_UPLOAD	0x02	Requests data transfer from device to host in order to load content of device internal flash memory into a host file
DFU_GETSTATUS	0x03	Requests device to send status report to the host
DFU_CLRSTATUS	0x04	Requests device to clear error status
DFU_GETSTATE	0x05	Requests the device to send only the state it enters immediately after this request
DFU_ABORT	0x06	Requests device to exit the current state and enter idle state immediately

Table 2 DFU class-specific requests

bmRequest	bRequest	wValue	wIndex	wLength	Data
00100001b	DFU_DETACH	wTimeout	interface	Zero	None
00100001b	DFU_DNLOAD	wBlockNum	interface	Length	Firmware
10100001b	DFU_UPLOAD	Zero	interface	Length	Firmware
00100001b	DFU_GETSTATUS	Zero	interface	6	Status
00100001b	DFU_CLRSTATUS	Zero	interface	Zero	None
00100001b	DFU_GETSTATE	Zero	interface	1	State
00100001b	DFU_ABORT	Zero	interface	Zero	None

2 DFU bootloader commands

The DFU_DNLOAD and DFU_UPLOAD requests are mainly used to perform write and read memory operations. They are also used to initiate the integrated bootloader commands (such as Access protect, erase/program protect, erase). The DFU_GETSTATUS command then triggers the command execution.

In the DFU_DNLOAD request, the command is selected through the wValue parameter in the USB request structure. If wValue=0, the data sent by the host after the request is a bootloader command code. The first byte is the command code.

In the DFU_UPLOAD request, the command is selected through the wValue parameter in the USB request structure. If wValue=0, the data sent by the host after the request is a bootloader command code. The first byte is the command code.

Table 3 DFU bootloader commands

DFU request	Bootloader command	Code	Description	Supported product series
DFU_UPLOAD	Read Memory	NA	Read data of the memory starting from an address specified by the application	Bootloader supports DFU
	Get	NA	Get Bootloader information including supported commands, project ID, product ID and version code	Bootloader supports DFU
	Get CRC	0xAC	Get CRC value of a sector	Bootloader supports DFU
	Get sLib Status	0xD2	Get sLib Status	Bootloader supports DFU and sLib
DFU_DNLOAD	Write Memory	NA	Write data to memory starting from an address specified by the application	Bootloader supports DFU
	Set Address	0x21	Set read/write address	Bootloader supports DFU
	Access Protect	0x82	Enable access protection	Bootloader supports DFU
	Access Unprotect	0x92	Disable access protection	Bootloader supports DFU
	Erase and Program Protect	0x63	Enable erase/program protection	Bootloader supports DFU
	Erase and Program	0x73	Disable erase/program	Bootloader supports DFU

DFU request	Bootloader command	Code	Description	Supported product series
	Unprotect		protection	
	Erase	0x41	Erase memory	Bootloader supports DFU
	Jump	0x18	Jump to memory address executable application	Bootloader supports DFU
	Firmware CRC	0xAC	Compute a CRC value on a given sector	Bootloader supports DFU
	Enable SPIM	0xB3	Enable SPIM	Bootloader supports DFU and SPIM
	Enable sLib	0xD0	Enable sLib	Bootloader supports DFU and sLib
	Disable sLib	0xD1	Disable sLib	Bootloader supports DFU and sLib
	Reset Device	0xD4	Reset Device	Only AT32F403xx does not support
	Advanced Access Protect	0xD6	Enable Advanced Access Protect	Bootloader supports DFU and Advanced Access Protect

3 DFU_UPLOAD request commands

The upload request is selected through the value of wValue in the USB request structure.

Figure 1 DFU_UPLOAD request: device side

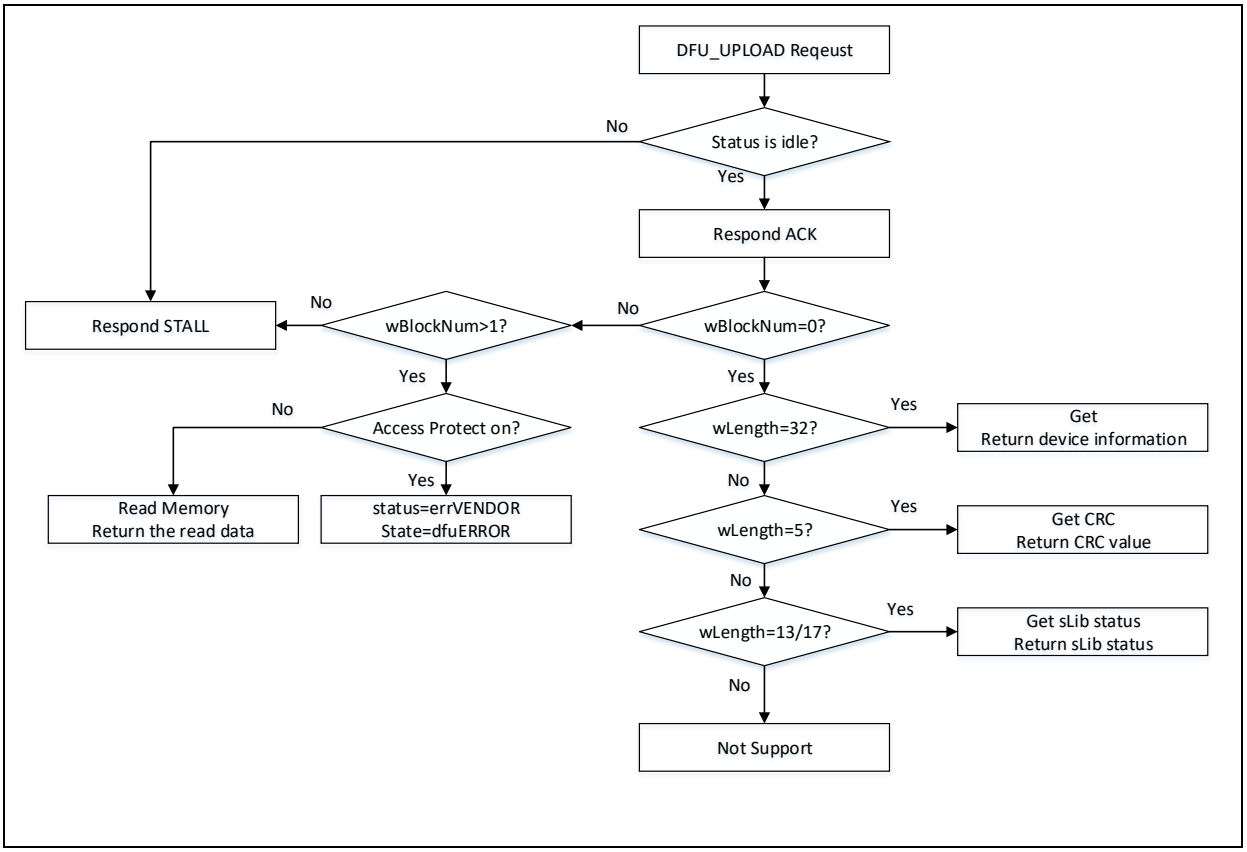
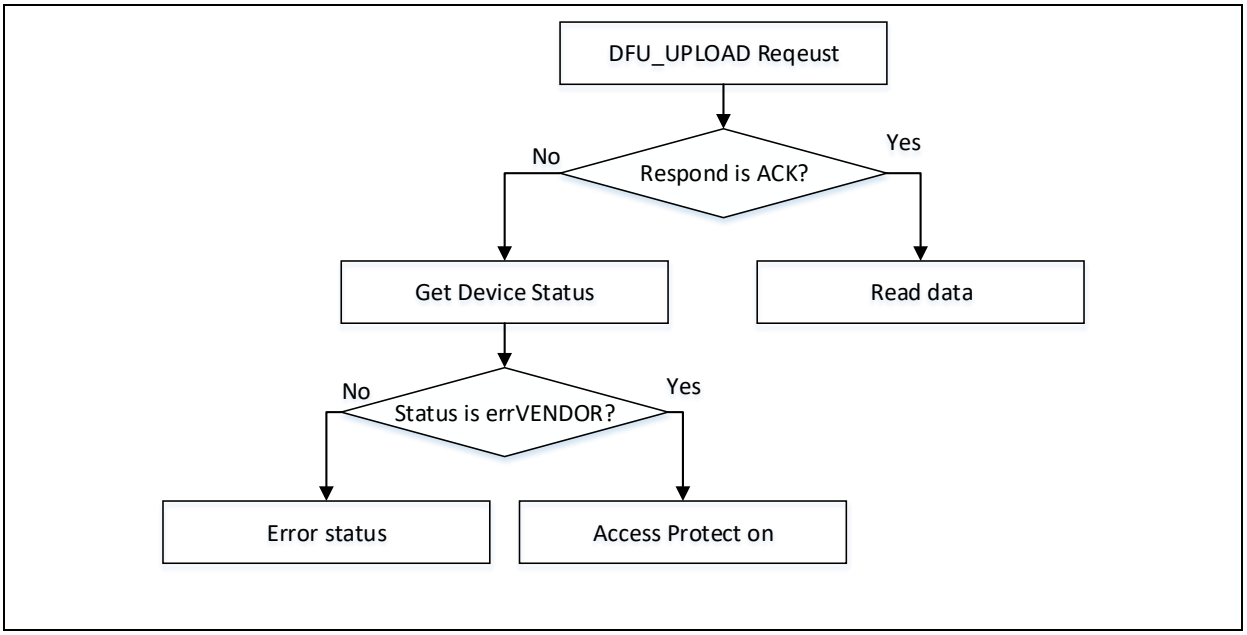


Figure 2 DFU_UPLOAD request: host side



3.1 Get command

This command is selected by setting wValue=0 and wLength=32. It is still active when the access protection is enabled.

This command is used to get the supported commands and basic information of the device.
The 12th byte to 16th byte represents the product ID and project ID of the device. The host gets the microcontroller ID by reading the Product ID and Project ID.
Note: the supported commands depend on the product series. The values returned from Get command are also dependent on the product series.

Table 4 Data received after Get command

Byte	Value	Description
0	0x00	Command
1	0x21	Set Address
2	0x41	Erase
3	0x92	Access Unprotect
4	0x82	Access Protect
5	0x63	Erase and Program protect
6	0x73	Erase and Program unprotect
7	0x18	Jump
8	0xAC	Firmware CRC
9	0xFF	Reserve
10	0x32	Protocol Version
11	0x00	Reserve
12	*	Product ID (LSB)
13	*	Product ID
14	*	Product ID
15	*	Product ID (MSB)
16	*	Project ID
17	*	Bootloader ID
18	*	Bootloader ID
19	0xD0	Enable sLib（if not support reserve）
20	0xD1	Disable sLib（if not support reserve）
21	0xD2	Get sLib status（if not support reserve）
22	0xB3	Enable SPIM（if not support reserve）
23	*	Reserve
24	0xD4	Reset Device（if not support reserve）
25	*	Reserve
26	0xD6	Advanced Access Protect（if not support reserve）
27~31	*	Reserve

3.2 Read memory

The Read memory operation is selected when wValue > 1. This command is inactive when the access protection is enabled.

The host requests the device to send a specified number of data types (wLength) from valid memory address of the internal flash memory, embedded SRAM and user system data area.

- For the internal flash memory, the embedded SRAM, read size can be 2 to 2048.
- For the user system data area, read size must be equal to the user system data area size

Note: the valid address range depends on the product series.

Read memory address is calculated according to the following formula:

Read initial address= (wValue – 2)* wlength + Address_Pointer

Address_Pointer is configured using “Set Address” command through the DFU_DNLOAD request.

If the access protection is enabled, the Read memory operation is not performed and the returned device status is Status=dfuERROR, state=errVENDOR.

3.3 Get CRC

This command is selected by setting wValue = 0 and wLength = 5. It is still active when the access protection is enabled.

The “Get CRC” command is used to check the integrity of firmware. The CRC value is computed based on a sector level. The host uses the Firmware CRC command initiated through the DFU_DNLOAD request to configure the start address and the number of sectors to be computed. After the computing operation, the host uses the Get CRC command initiated through the DFU_UPLOAD request to get the computed CRC values.

Data returned after Get CRC command

Byte	Value	Description
0	0xAC	Get CRC
1	*	CRC (LSB)
2	*	CRC
3	*	CRC
4	*	CRC (MSB)

3.4 Get sLib status

This command is selected by setting wValue = 0 and wLength = 13/17. It is inactive when the access protection is enabled.

The Get sLib status command is used to get the current sLib status. After the command operation, the device returns the sLib register values. For more details concerning sLib register, refer to the user reference manual of the corresponding product series.

AT32F435xx/AT32F437xx: (return 17 bytes data)
When the device receives the command, it sends the values of the SLIB_STS0 register (four bytes), the SLIB_STS1 register (four bytes), the SLIB_STS2 register (four bytes) and the SLIB_MISC_STS register (four bytes).

Other product series (return 13 bytes data)
When the device receives the command, it sends the values of the SLIB_STS0 register (four bytes), the SLIB_STS1 register (four bytes), the SLIB_STS2 register (four bytes) and the SLIB_MISC_STS register (four bytes).

Note: For AT32F403A, AT32F407 and AT32F413 series, this command cannot be used when the access protection is enabled. For other product series, this command is still active when the access protection is enabled.

AT32F435xx/AT32F437xx returned data

Byte	Value	Description
0	0xD2	Get sLib Status
1	*	SLIB_STS0（LSB）
2	*	SLIB_STS0
3	*	SLIB_STS0
4	*	SLIB_STS0（MSB）
5		SLIB_STS1（LSB）
6		SLIB_STS1
7		SLIB_STS1
8		SLIB_STS1（MSB）
9		SLIB_STS2（LSB）
10		SLIB_STS2
11		SLIB_STS2
12		SLIB_STS2（MSB）
13		SLIB_MISC_STS（LSB）
14		SLIB_MISC_STS
15		SLIB_MISC_STS
16		SLIB_MISC_STS（MSB）

Data returned for other product series:

Byte	Value	Description
0	0xD2	Get sLib Status
1	*	SLIB_STS0（LSB）
2	*	SLIB_STS0
3	*	SLIB_STS0
4	*	SLIB_STS0（MSB）
5	*	SLIB_STS1（LSB）
6	*	SLIB_STS1
7	*	SLIB_STS1
8	*	SLIB_STS1（MSB）
9	*	SLIB_MISC_STS（LSB）
10	*	SLIB_MISC_STS
11	*	SLIB_MISC_STS
12	*	SLIB_MISC_STS（MSB）

4 DFU_DNLOAD requests

The DFU_DNLOAD command is selected through the value of the wValue and the first byte of the data.

Table 5 DNLOAD command conditions

Command	Byte0 Value	wValue
Write Memory	*	>1
Set Address	0x21	0
Access Protect	0x82	0
Access Unprotect	0x92	0
Erase and Program protect	0x63	0
Erase and Program unprotect	0x73	0
Erase	0x41	0
Jump	0x18	0
Firmware CRC	0xAC	0
Enable sLib	0xD0	0
Disable sLib	0xD1	0
Reset Device	0xD4	0
Advanced Access Protect	0xD6	0

Note: Before issuing a DFU_DNLOAD request, the host must check that the derive is in a correct state (dfuIDLE or dfuDNLOAD-IDLE). If the device is not in idle state, the host must send the DFU_CLRSTATUS request and get the status again until the device returns to the dfuIDLE state.

Figure 3 DFU_DNLOAD request: device side

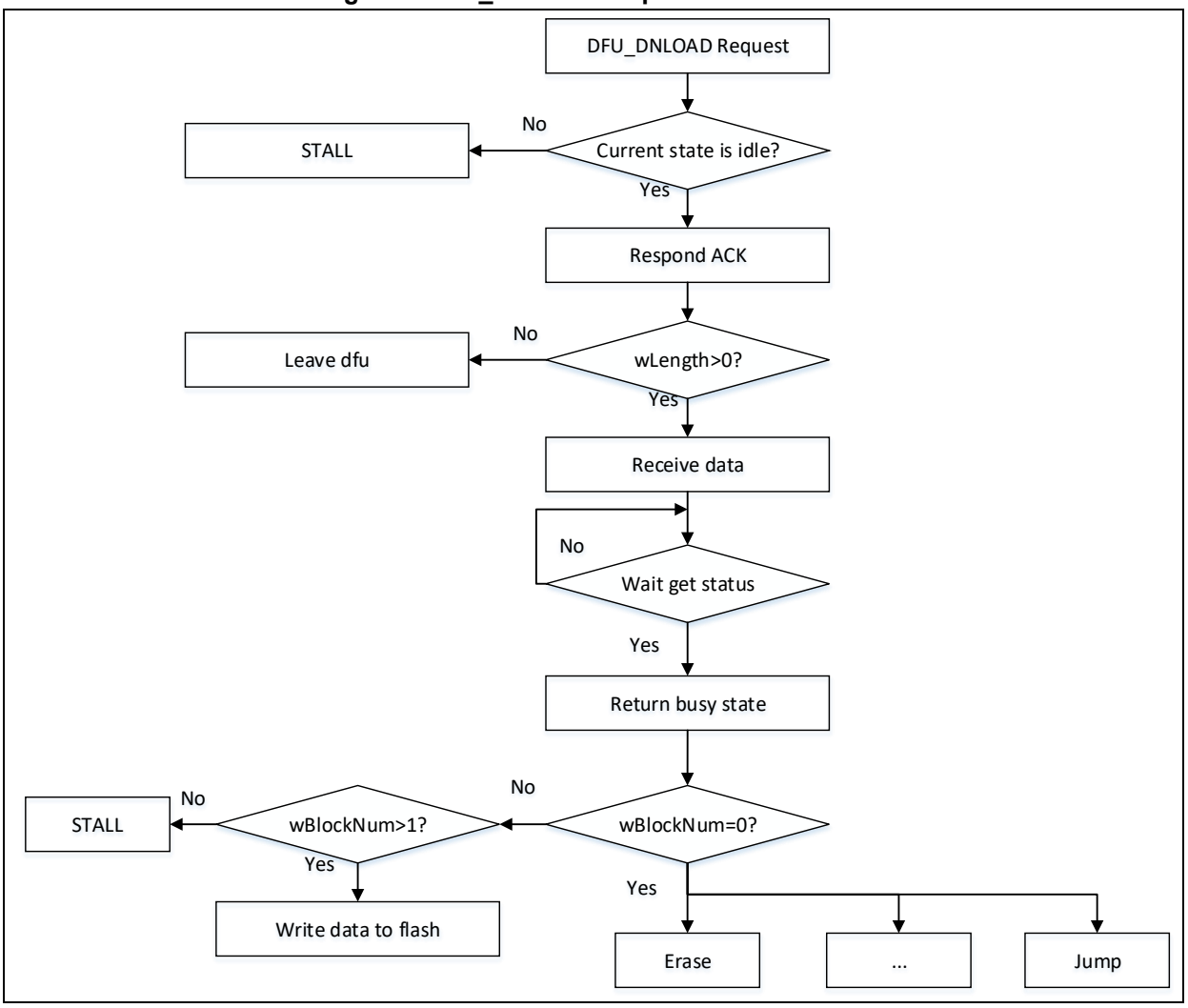
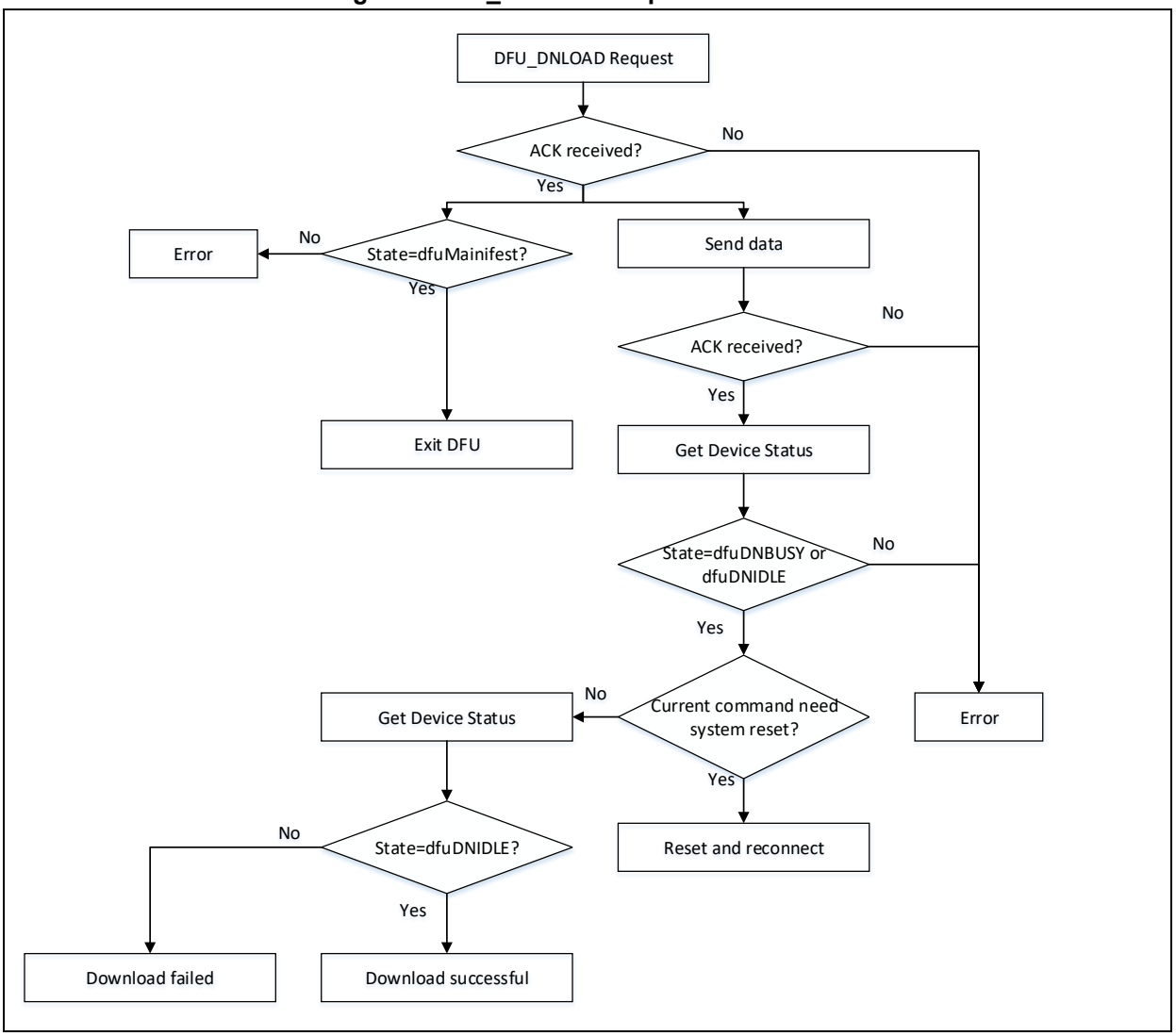


Figure 4 DFU_DNLOAD request: host side



4.1 Write Memory

This command is selected when wValue > 1. This command is not performed when the access protection is enabled.

The host sends a valid address (through “Set Address” command) and a valid data length (wLength) to the device. The device writes data to the specified address.

The allowed number of bytes to write depends upon the target memory:

- For the internal flash memory and embedded SRAM, the write size ranges from 2 to 2048 bytes
- For the user system data area, the write size must be equal to the user system data area size

Note: the valid address range depends on the product series. Refer to the product series reference manual for more details concerning this size.

The Write memory operation is effectively executed only when a DFU_GETSTATUS request is issued by the host. Then the status returned by the device is dfuDNBUSY (it indicates that write memory is going on). If the Write memory command is issued to the user system data area, the device generates a system reset at the end of the command. At the same time, the host must be

reconnected to the device.

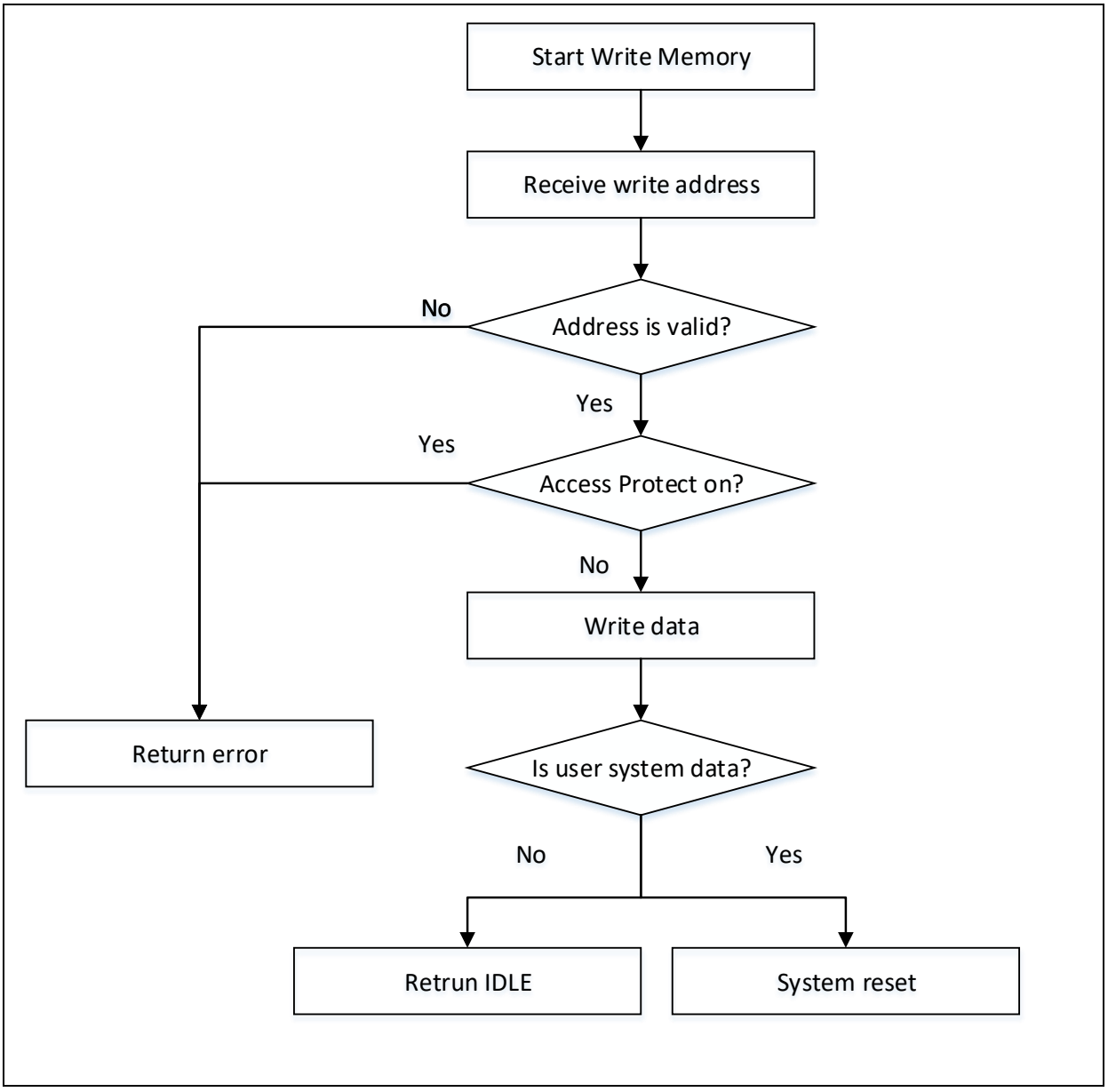
A second DFU_GETSTATUS request is needed to check if the command has been correctly executed. If the received address is wrong or unsupported, the device status is status=dfuERROR, state =STATUS_ERRADDRESS.

The address where the host requests to write data is computed using the following formula:

Initial address =(wValue – 2)* wlength + Address_Pointer

If the access protection is enabled, the status returned is status=dfuERROR, state=errVENDOR

Figure 5 Write Memory: device side



4.2 Set Address

This command is selected when wValue=0 and the first byte value sent by the host is 0x21. The command is not performed when the access protection is enabled.

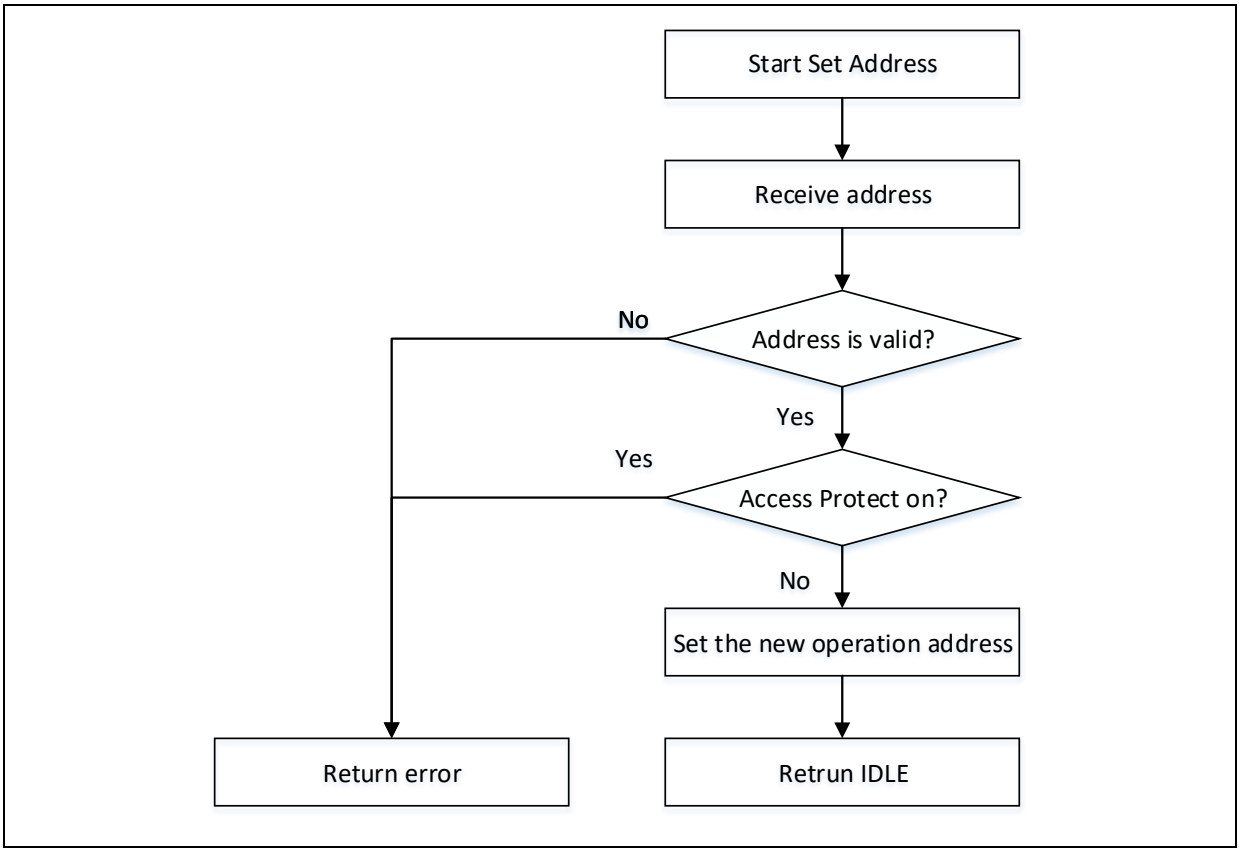
Before starting “Write memory” and “Read memory” commands, the “Set Address” command must be issued to set the address pointer used for computing the start address for Read and Write memory operations. The host sends “Set Address” command (one byte) and address (four bytes).

After sending the Set Address command, the host must send the DFU_GETSTATUS request. The Set Address command is executed only when a DFU_GETSTATUS request is issued by the host. The status returned by the device is STATE_dfuDNLOAD_IDLE (Set address command completion)

Data sent from the host

Byte	Value	Description
0	0x21	Set Address command
1	*	Address (LSB)
2	*	Address
3	*	Address
4	*	Address (MSB)

Figure 6 Set Address: device side



4.3 Access Protect

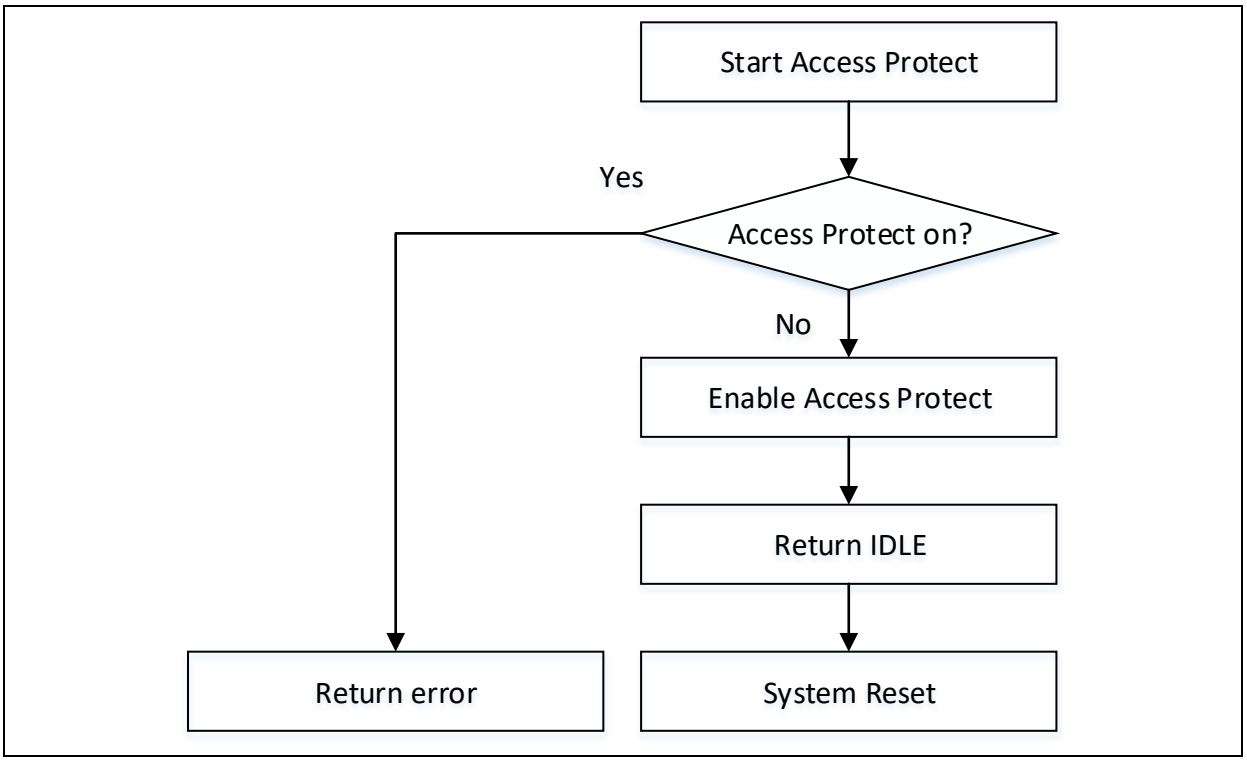
This command is selected when wValue=0 and the first byte value sent by the host is 0x82. The command is not allowed and executed when the access protection is enabled.

After sending the access protection enable command (0x82), the host must send a DFU_GETSTATUS request to trigger the execution of the command by the device. Then the status returned by the device is STATE_dfuDNLOAD_IDLE. After this operation, the device performs a system reset for the access protection command to take effect. At the same time, the host must be reconnected to the device.

Data sent by the host:

Byte	Value	Description
0	0x82	Access Protect

Figure 7 Access Protect: device side



4.4 Access Unprotect

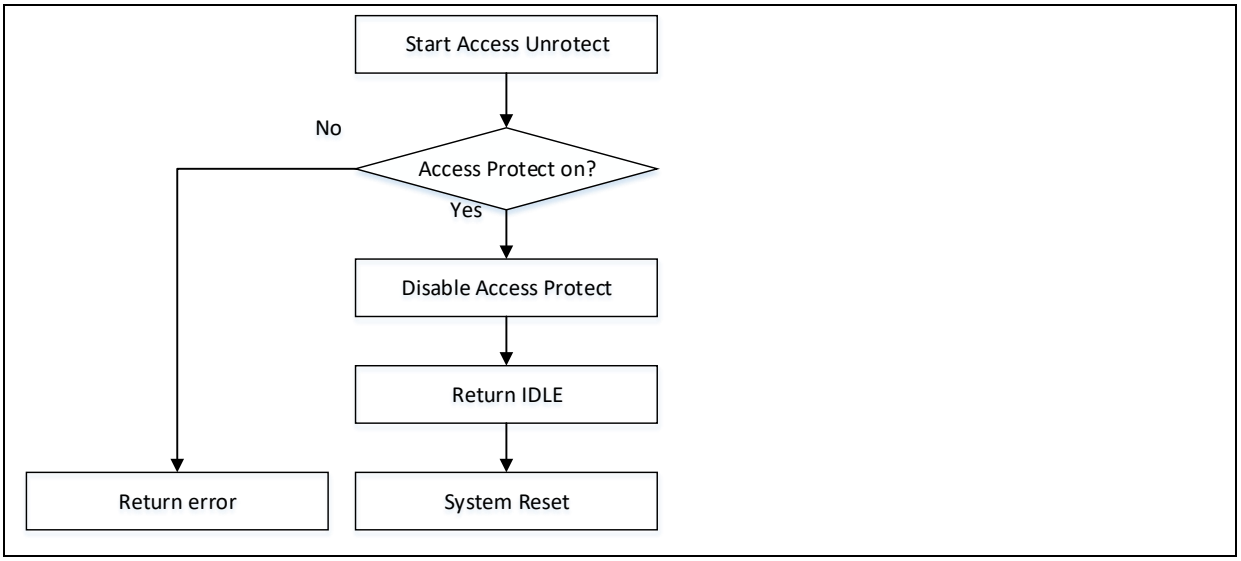
This command is selected when wValue=0 and the first byte value sent by the host is 0x92. The command is allowed and executed when the access protection is enabled.

After sending a Read Unprotect command, the host must send a DFU_GETSTATUS request. After this operation, the device removes the access protection. At the same time, the device returns STATE_dfuDNLOAD_IDLE. Then the main flash memory is fully erased. After executing this command, the device executes a system reset.

Data sent by the host:

Byte	Value	Description
0	0x92	Access Unprotect

Figure 8 Access Unprotect: device side



4.5 Erase and Program Protect command

This command is selected when wValue=0 and the first byte value sent by the host is 0x63. The command is not allowed and performed when the access protection is enabled.

The host sends the Erase and program protect command (one byte), the number of sector index (one byte) --- for AT32F435/437 series, the number of sectors to send is the actual number of sectors – 1; for other series, it is the number of sectors to send--- and sector index (n bytes). After this operation, the device writes the user system data area based on the number of sectors and sector index sent by the host.

After sending the Write protect command, the host must send a DFU_GETSTATUS request. Then the device enables the write protection. After this operation, the device executes a system reset.

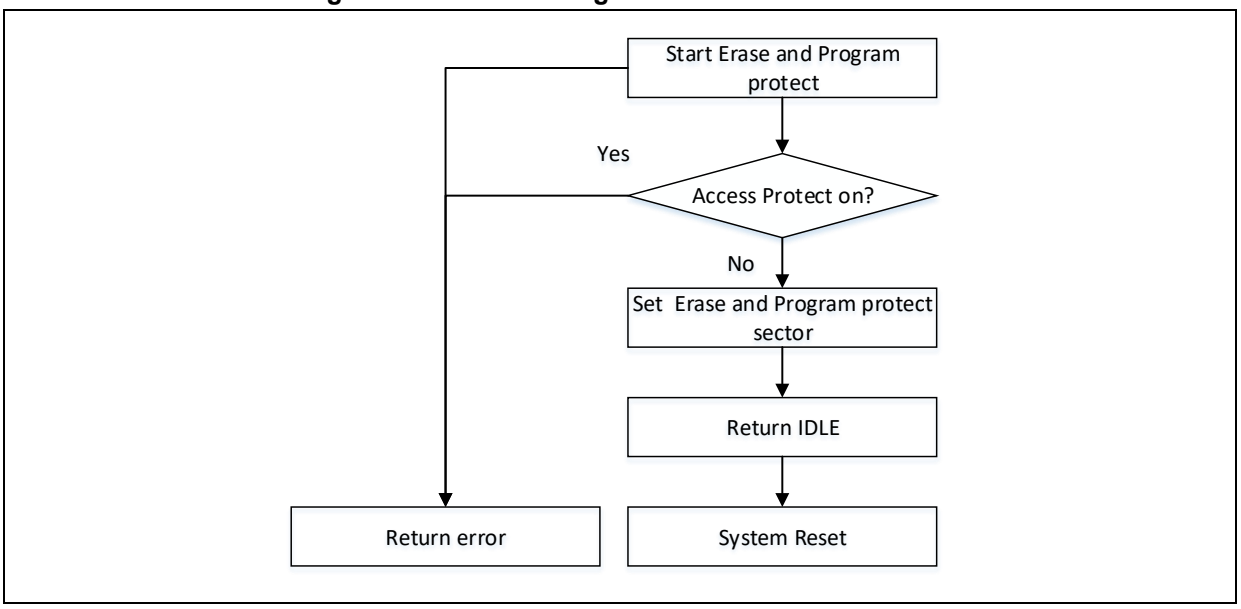
Note: The index bit (value 0,1,2…n) in the Erase/Program protect command corresponds to the bit (0-N) of the Erase/Program protect byte in the user system data area. Refer to the corresponding product reference manual for more details concerning this issue.

Note: For AT32F435/437 series, the number of sector index to send is the actual value -1. For other product series, the number of sector index to send is equal to that of sectors to send.

Data sent by the host:

Byte	Value	Description
0	0x63	Erase and Program Protect
1	*	Number of sector index(435/437 need - 1)
2	*	Sector index 0
3	*	Sector index1
4	…	…

Figure 9 Erase and Program Protect: device side



4.6 Erase and Program Unprotect

This command is selected when wValue=0 and the first byte value sent by the host is 0x73. The command is not allowed and executed when the access protection is enabled.

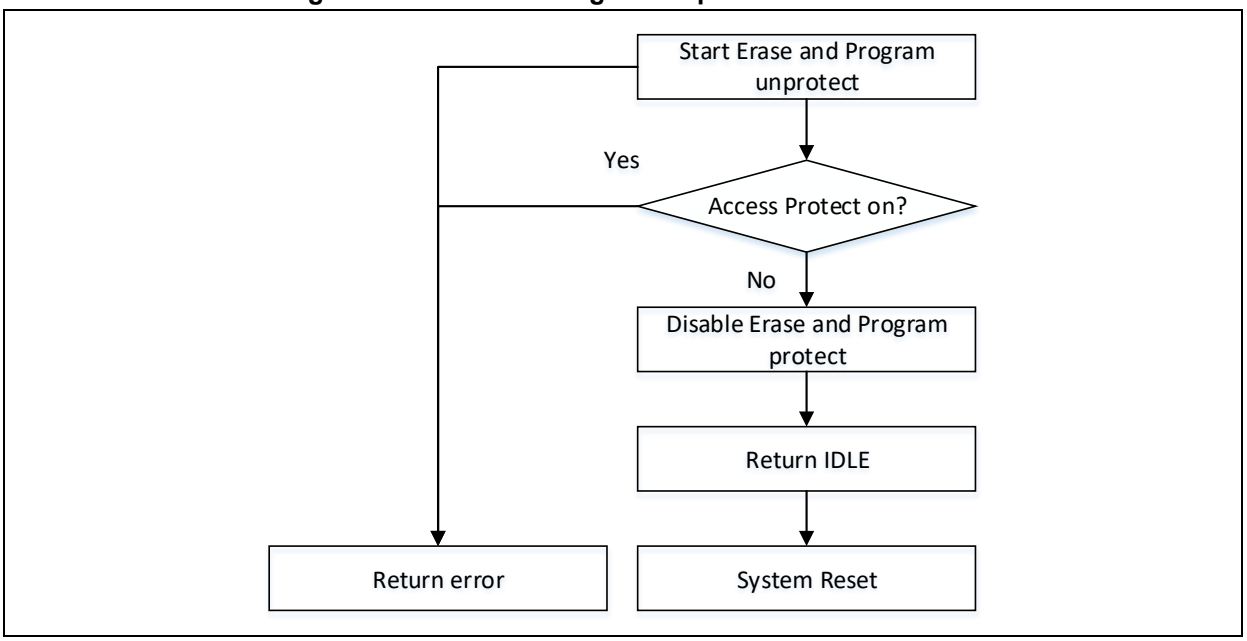
This command is used to remove Erase/program protection of all sectors. The host only needs to send a byte Erase/Program unprotect ion command.

After sending the Erase/Program unprotect command, the host must send a DFU_GETSTATUS request. Then the device removes the Erase/Program protection operation. After this operation, the device returns STATE_dfuDNLOAD_IDLE and then executes a system reset.

Data sent by the host:

Byte	Value	Description
0	0x73	Erase and Program Unprotect

Figure 10 Erase and Program Unprotect: device side



4.7 Erase command

This command is selected when wValue=0 and the first byte value sent by the host is 0x41. The command is not allowed and executed when the access protection is enabled.

This command supports sector erase (sector size depends on the product series) and mass erase. For those products that contain bank 1 and bank 2, this command can be used to erase bank 1 and bank 2.

For those with SPIM feature, it support erase operation on Bank 3.

Note that this command is used to erase any valid memory address, depending on the product series.

- Mass erase: wLength=1, only a 1-byte Erase command is needed
- Sector erase: wLength=5, the host needs to send a 1-byte erase command a four-byte erase address (aligned with sectors)
- Bank erase: wLength=3, the host needs to send a 1-byte erase command and a 2-byte Bank index
- Block erase: wLength=7, the host needs to send a –byte erase command, a 2-byte block index and a 4-byte erase address

Note: The AT32F435/AT32F437 series supports block erase. And the block erase must be operated on the basis of 64kbytes.

Table 6 Erase index

Erase type	Erase index	Erase area
Mass erase	0xFF 0xFF	All Flash
Bank erase	0xFF 0xFE	Bank1 Erase
	0xFF 0xFD	Bank2 Erase
	0xFF 0xFC	Bank3 Erase
Block erase	0xFF 0xFB	Block Erase

After sending an Erase command, the host must send a DFU_GETSTATUS request. If the status returned by the device is not dfuDNBUSY, it indicates a wrong or invalid address or the access protection is enabled. The device gets the current status through the DFU_GETSTATUS request until the returned status is dfuDNLOAD_IDLE, which indicates the end of the erase command.

Mass erase: data sent by the host (wLength=1)

Byte	Value	Description
0	0x41	Erase
1	*	Number of sector index
2	*	Sector index 0
3	*	Sector index1
4	…	…

Sector erase: data sent by the host (wLength=5):

Byte	Value	Description
0	0x41	Erase
1	*	Address (LSB)
2	*	Address

Byte	Value	Description
3	*	Address
4	*	Address
5	*	Address (MSB)

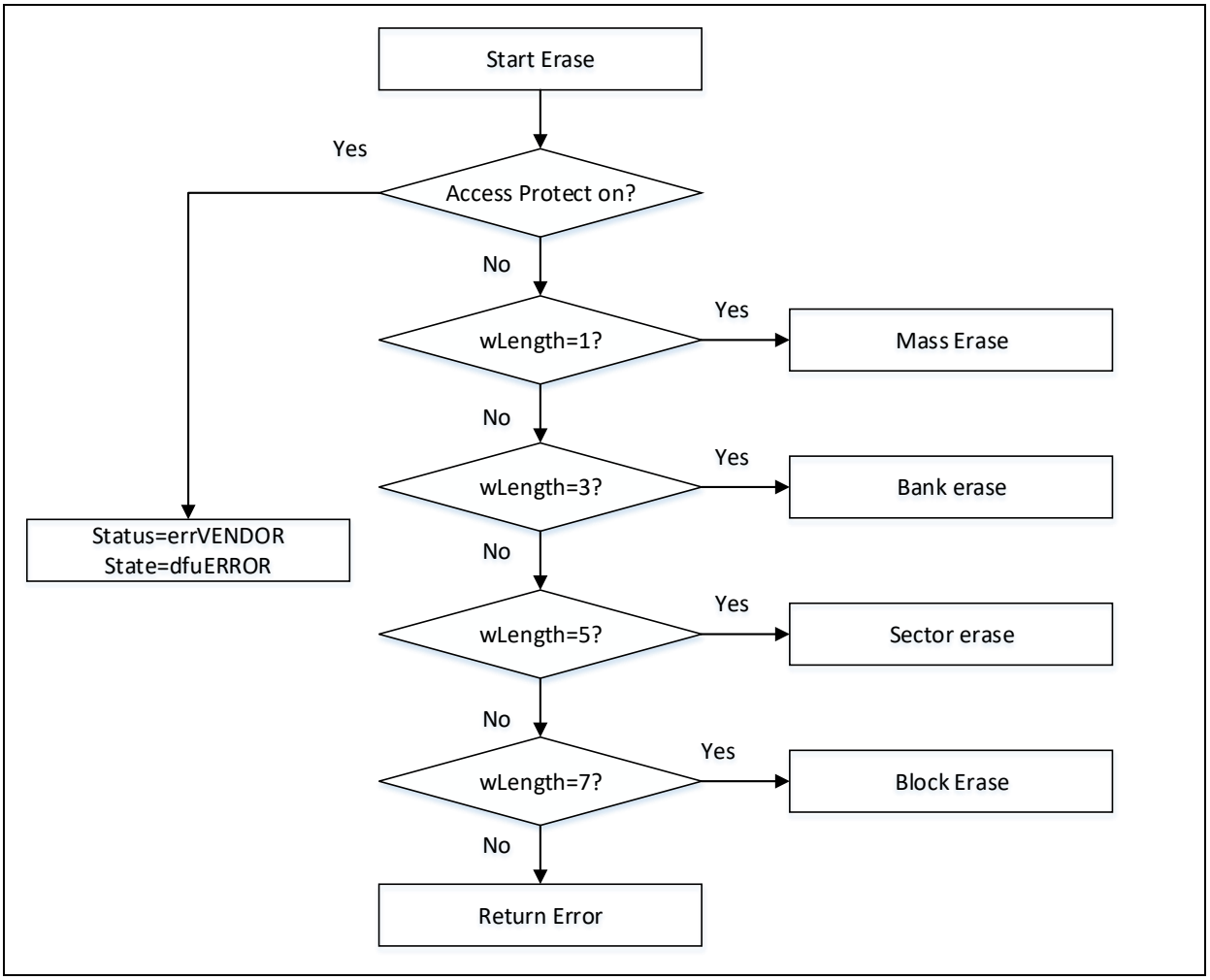
Bank erase: data sent by the host (wLength=3)

Byte	Value	Description
0	0x41	Erase
1	0xFF	Bank index
2	*	Bank index (0XFE 0xFD 0xFC)

Block erase: data sent by the host (wLength=7)

Byte	Value	Description
0	0x41	Erase
1	0xFF	Block index
2	0xFB	Block index
3		Address (LSB)
4	*	Address
5	*	Address
6	*	Address (MSB)

Figure 11 Erase: device side



4.8 Jump command

This command is selected when wValue=0 and the first byte value sent by the host is 0x18. The command is not allowed and executed when the access protection is enabled.

This command is used to jump to a specified address to execute code. The host needs to send a 1-byte Jump command and a 4-byte valid Jump address.

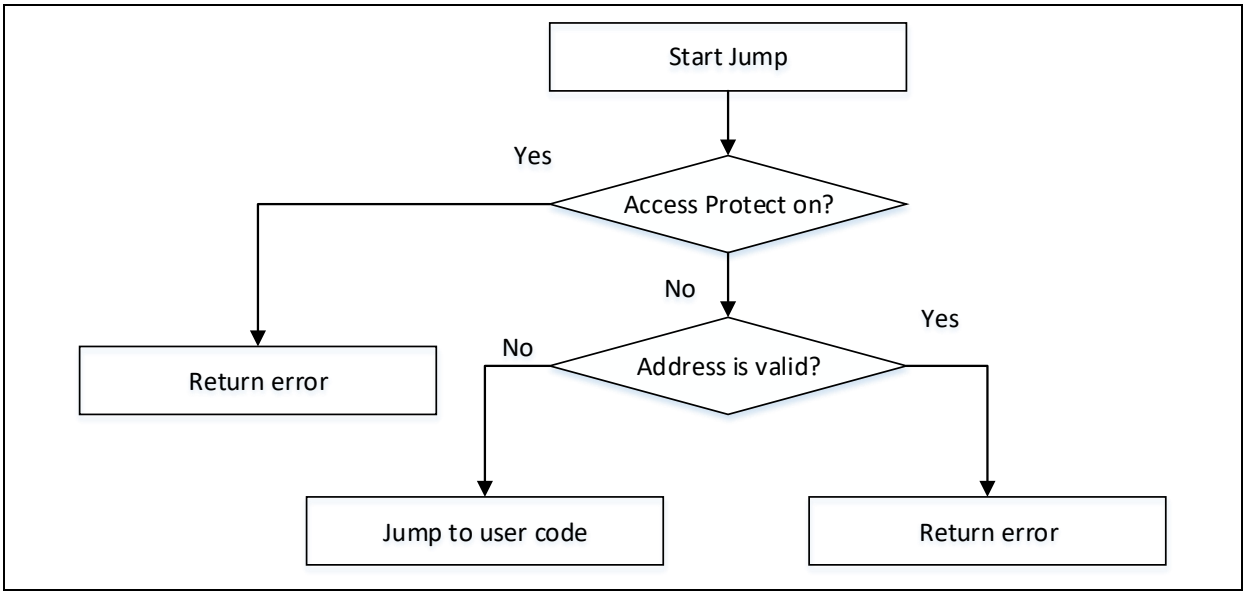
After sending the Jump command, the host must send a DFU_GETSTATUS request. After this operation, the device returns dfuDNLOAD_IDLE, indicating it is ready to Jump to user code execution.

Note: The valid address range of SRMA, flash memory, SPIM(BANK3) depends on the product series. Refer to the corresponding product reference manual for more details.

Data sent by the host:

Byte	Value	Description
0	0x18	Jump
1	*	Address (LSB)
2	*	Address
3	*	Address
4	*	Address
5	*	Address (MSB)

Figure 12 Jump: device side



4.9 Firmware CRC command

This command is selected when wValue=0 and the first byte value sent by the host is 0xAC. The command is allowed and executed when the access protection is enabled.

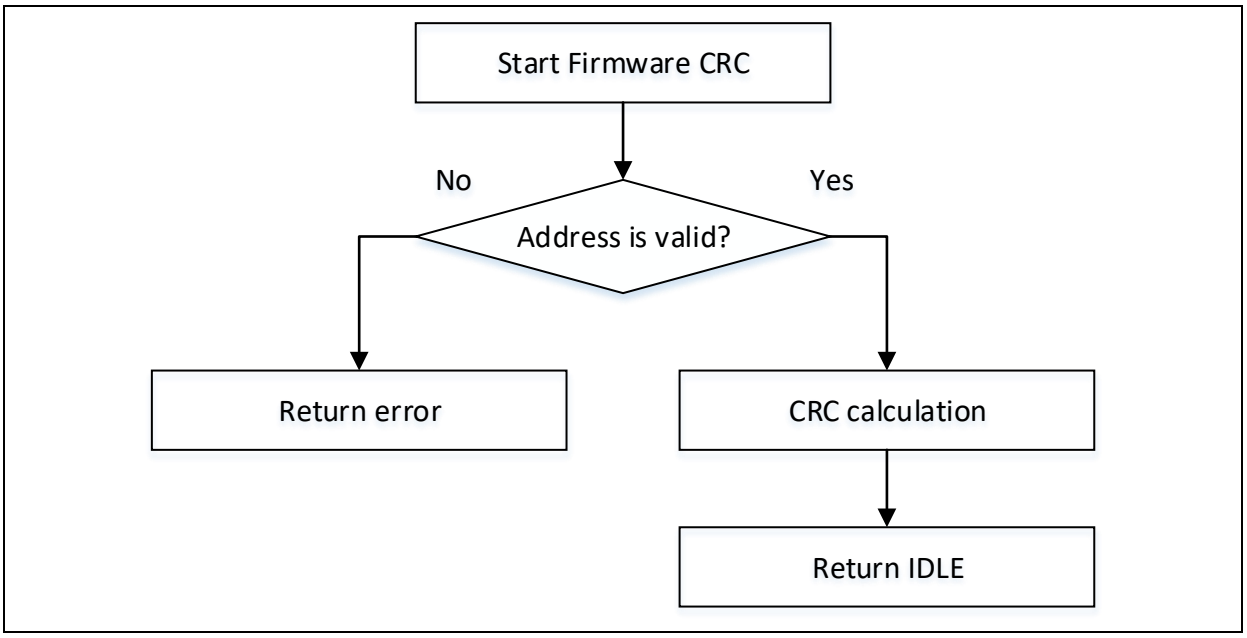
The CRC is computed based on a sector level. The host must send a 1-byte CRC compute command, a 4-byte start address (aligned with sectors) and a 2-byte sector count.

After sending this command, the host must send a DFU_GETSTATUS request. If the device returns dfuDNLOAD_IDLE, it indicates the completion of the CRC computing. Then a Get CRC command is issued through a DFU_UPLOAD request to get the current CRC value.

Data sent by the host:

Byte	Value	Description
0	0xAC	Erase
1	*	Address (LSB)
2	*	Address
3	*	Address
4	*	Address (MSB)
5	*	Number of sector (LSB)
6	*	Number of sector (MSB)

Figure 13 Firmware CRC: device side



4.10 Enable SPIM command

This command is selected when wValue=0 and the first byte value sent by the host is 0xB3. The command is allowed and executed when the access protection is enabled.

This command is used to enable bank 3. SPIM Flash type, Flash size and Flash memory encryption range shall be available for the device to use this command. Refer to the SPIM use manual for more details.

Flash type list:

- 0x90: general Flash
  Dummy cycle is 4
- 0x91: general Flash, and Quad Enable setting
  Dummy cycle is 4
  Quad Enable is activated in “Volatile” format.

The maximum allowable limit for the Flash size is 16M Bytes. The device erases, reads or writes the Flash size sent from the host.

In order to prevent SPIM Flash from being read out, SPIM is equipped with an encryption feature. In other words, it allows the Flash memory controller to write encrypted data to the SPIM, and the data are then decrypted when readout.

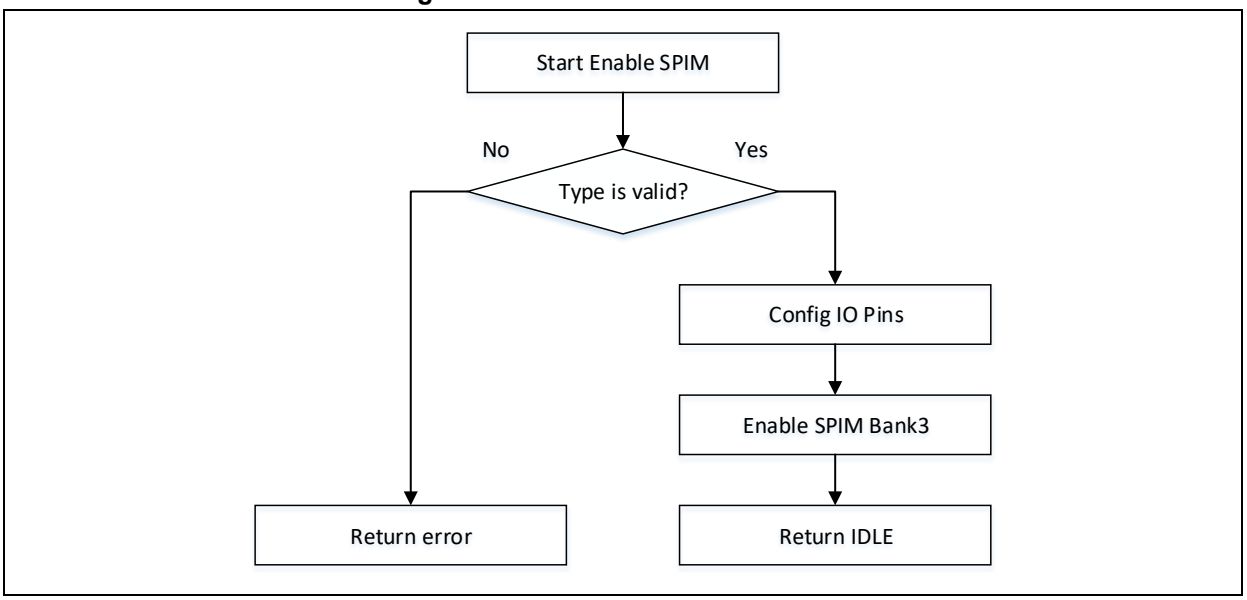
For MCU, read and write operation are done in a plaintext format. But those data which are stored in the external Flash memory are encrypted. The user can define a password and set encrypted range. The encryption key is located in the user system data area and must be set before write operation to SPIM (bank3).

Encryption range (also called FLASH_FDA) is configured before enabling SPIM (bank3) each time. It is used to define how many of the bytes starting from the SPIM (bank3) initial address need to be encrypted. When a value greater than 16M Bytes is set, it indicates that all of the SPIM (bank3) must be encrypted. When the value is 0, it indicates that the whole SPIM (bank3) is not encrypted

Data sent by the host

Byte	Value	Description
0	0xB3	Enable SPIM
1	*	Flash type
2	*	Flash size（LSB）
3	*	Flash size
4	*	Flash size
5	*	Flash size（MSB）
6	*	Flash FDA（LSB）
7	*	Flash FDA
8	*	Flash FDA
9	*	Flash FDA（MSB）

Figure 14 Enable SPIM: device side



4.11 Enable sLib

This command is selected when wValue=0 and the first byte value sent by the host is 0xD0. The command is not allowed and executed when the access protection is enabled.

To enable this command, the host must send a 4-byte sLib password, 2-byte sLib start sector, a 2-byte sLib data/instruction start sector and a 2-byte sLib end sector. The settings of the sLib must be taken into account only after a system reset.

After sending the Enable sLib command, the host must send a DFU_GETSTATUS request. Then the device enables sLib. However, the configuration of the sLib must take effect only after a reset. Thus a reset device command must be issued following the Enable sLib command and data download to memory.

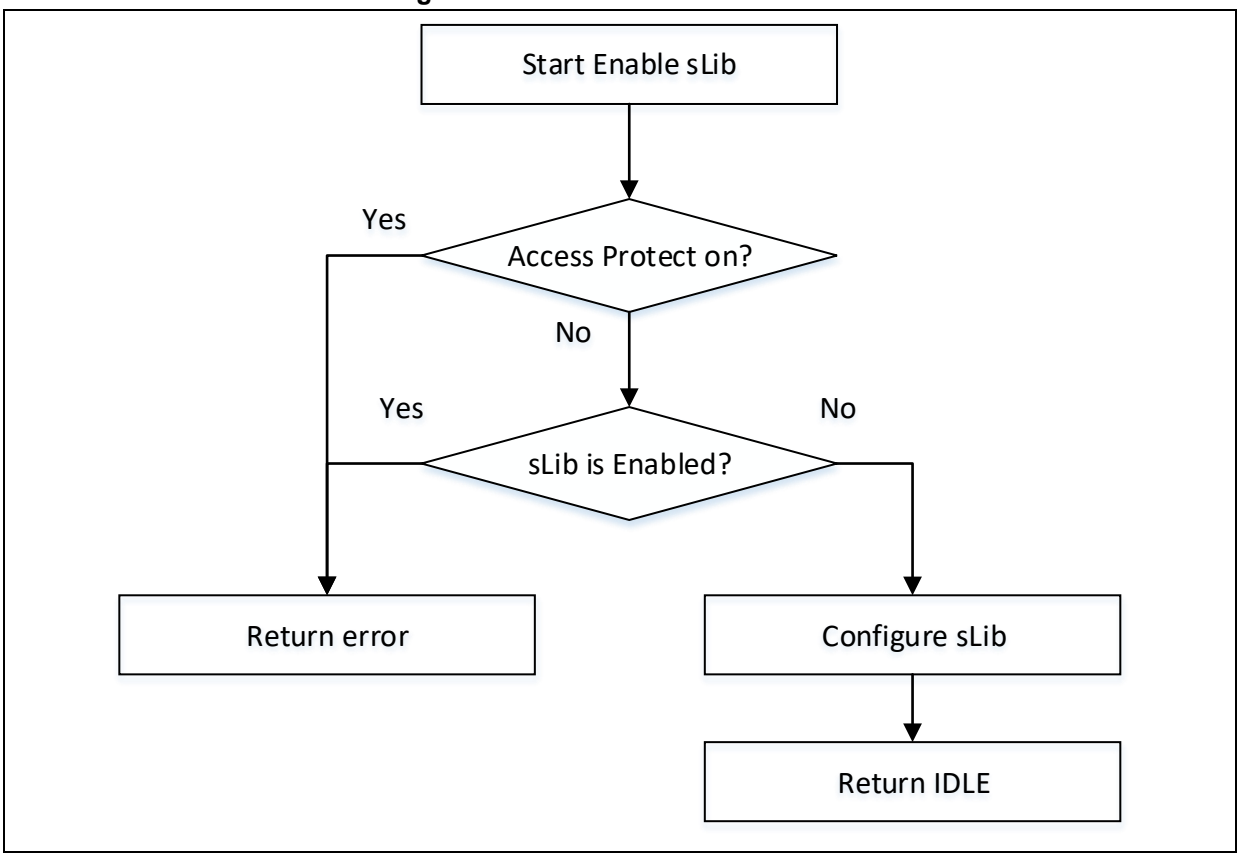
Note: 1. Refer to the security library user guide for more details.

2. The sLib status shall be obtained before enabling sLib operation. If sLib is enabled in advance, it is necessary to disable sLib and then enable it.

Data sent by the host:

Byte	Value	Description
0	0xD0	Enable sLib
1	*	Password（LSB）
2	*	Password
3	*	Password
4	*	Password（MSB）
5	*	sLib start sector（LSB）
6	*	sLib start sector（MSB）
7	*	sLib data/instruction start sector（LSB）
8	*	sLib data/instruction start sector（MSB）
9	*	sLib end sector（LSB）
10	*	sLib end sector（MSB）

Figure 15 Enable sLib: device side



4.12 Disable sLib

This command is selected when wValue=0 and the first byte value sent by the host is 0xD1. The command is not allowed and executed when the access protection is enabled.

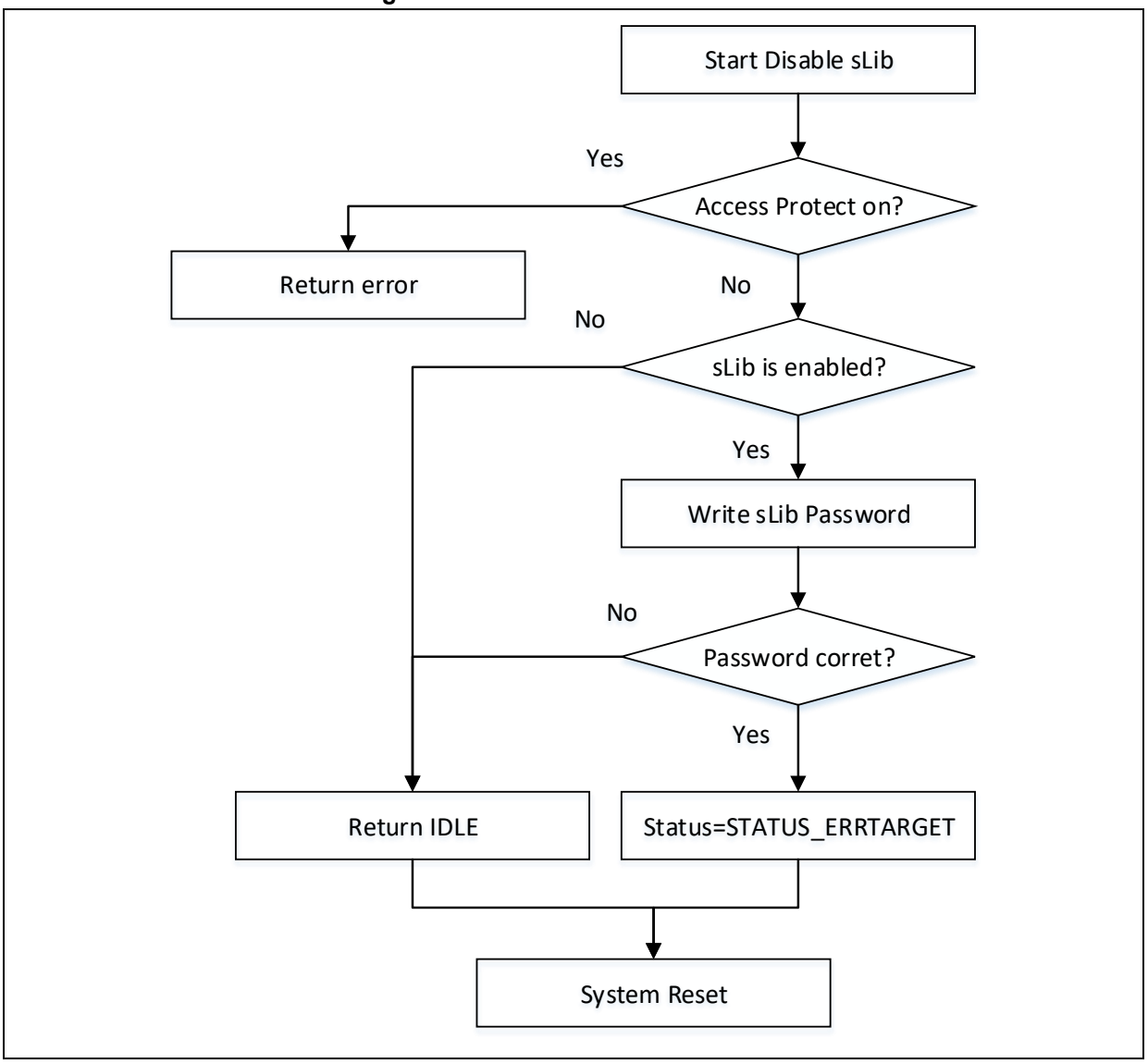
To issue Disable sLib command, the host must send a 1-byte Disable sLib command and a 4-byte password. If the password is correct, the device disables sLib and executes a system reset. If the password is wrong, the device returns an error status= STATUS_ERRTARGET

After sending the Disable sLib command, the host must send a DFU_GETSTATUS request. The device disables sLib. After activating the Disable sLib command, the device executes a system reset automatically whatever the password. And the host must be reconnected.

Data sent by the host:

Byte	Value	Description
0	0xD1	Disable sLib
1	*	Password（LSB）
2	*	Password
3	*	Password
4	*	Password（MSB）

Figure 16 Disable sLib: device side



4.13 Reset Device

This command is selected when wValue=0 and the first byte value sent by the host is 0xD4. The command is not allowed and executed when the access protection is enabled.

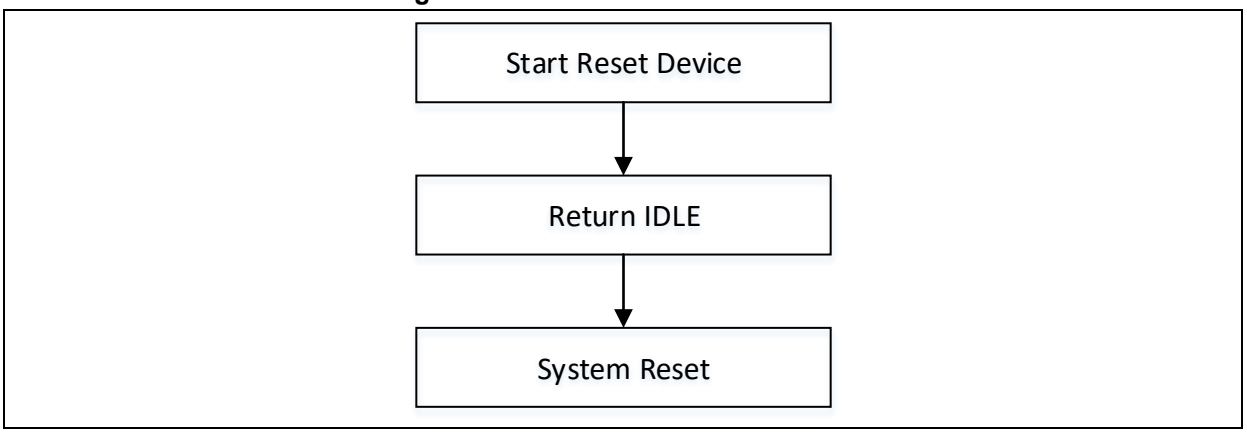
This command is used to reset a device following certain operations performed by the host.

After sending the Reset Device command, the host must send a DFU_GETSTATUS request. Then the device performs reset operation.

Data sent by the host:

Byte	Value	Description
0	0xD4	Reset Device

Figure 17 Reset Device: device side



4.14 Advanced Access Protect

This command is selected when wValue=0 and the first byte value sent by the host is 0xD6. The command is not allowed and executed when the access protection is enabled.

Refer to the corresponding product reference manual for more details concerning the advanced access protection.

Note: the advanced access protection cannot be removed for some product series. Refer to the corresponding product reference manual for more details concerning the advanced access protection.

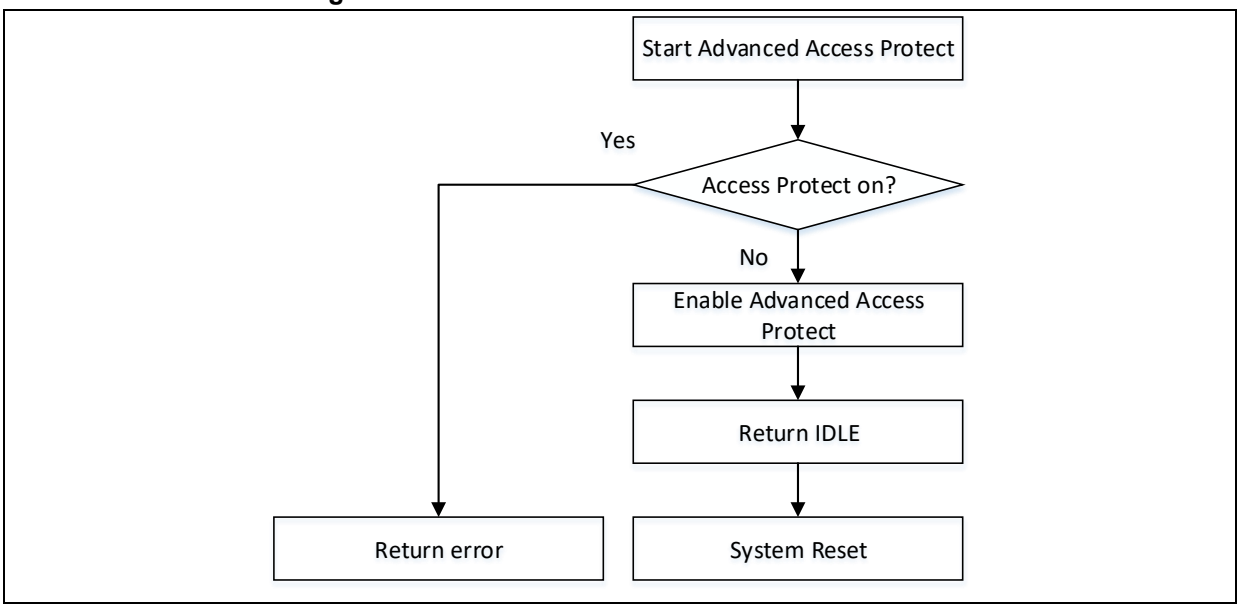
To enable this command, the host must send a 1-byte Advanced Access Protection command and a 1-byte flag.

After sending the Advanced Access Protection command, the host must send a DFU_GETSTATUS request. Then the device enables the advanced access protection before executing a system reset. At the same time, the host must be reconnected.

Data sent by the host

Byte	Value	Description
0	0xD6	Advanced Access Protect
1	0x02	Flag

Figure 18 Advanced Access Protect: device side



5 Revision history

Table 7 Document revision history

Date	Revision	Changes
2021.12.07	2.0.0	Initial release
2023.01.05	2.0.1	1. Add AT32F423 series 2. Revise the description about the number of sectors in the “Erase and Program Protect” of the AT32F435/437 series.
2023.06.30	2.0.2	1. Add AT32F405/402 series 2. Add AT32A403A series
2024.03.20	2.0.3	1. Add AT32A423 series
2024.04.28	2.0.4	1. Update supported product list 2. Update supported command list

IMPORTANT NOTICE – PLEASE READ CAREFULLY

Purchasers are solely responsible for the selection and use of ARTERY's products and services, and ARTERY assumes no liability whatsoever relating to the choice, selection or use of the ARTERY products and services described herein

No license, express or implied, to any intellectual property rights is granted under this document. If any part of this document deals with any third party products or services, it shall not be deemed a license granted by ARTERY for the use of such third party products or services, or any intellectual property contained therein, or considered as a warranty regarding the use in any manner of such third party products or services or any intellectual property contained therein.

Unless otherwise specified in ARTERY's terms and conditions of sale, ARTERY provides no warranties, express or implied, regarding the use and/or sale of ARTERY products, including but not limited to any implied warranties of merchantability, fitness for a particular purpose (and their equivalents under the laws of any jurisdiction), or infringement on any patent, copyright or other intellectual property right.

Purchasers hereby agree that ARTERY's products are not designed or authorized for use in: (A) any application with special requirements of safety such as life support and active implantable device, or system with functional safety requirements; (B) any aircraft application; (C) any aerospace application or environment; (D) any weapon application, and/or (E) or other uses where the failure of the device or product could result in personal injury, death, property damage. Purchasers' unauthorized use of them in the aforementioned applications, even if with a written notice, is solely at purchasers' risk, and Purchasers are solely responsible for meeting all legal and regulatory requirements in such use.

Resale of ARTERY products with provisions different from the statements and/or technical characteristics stated in this document shall immediately void any warranty grant by ARTERY for ARTERY's products or services described herein and shall not create or expand any liability of ARTERY in any manner whatsoever.

© 2024 Artery Technology -All rights reserved