GigaDevice Semiconductor Inc.

Arm® Cortex®-M3/4/23/33 32-bit MCU

Application Note AN017



Table of Contents

Table of	Contents	2
List of F	igures	3
List of T	able	4
1. Intro	duction	5
2. Use	J-Flash SPI host computer to download files to SPI Nor Flash	6
2.1. H	Hardware connection	6
2.2. J	Jflash-SPI host computer configuration and download	6
3. Use l	KEIL to download files to SPI Nor Flash12	2
3.1. N	New FLM project12	2
3.2. F	Porting SPI Flash driver code12	2
3.3. N	Nodify FlashDevice structure1	5
3.4. 0	Compile and generate FLM file1	6
3.5. <i>I</i>	Add algorithm file to KEIL project1	7
3.6. 0	Compile and download1	8
3.7. 1	Testing and verification1	9
4. Revis	sion history20	0



List of Figures

Figure 2-1. GD25Q16BS schematic diagram (left) and JTAG pin diagram (right)	6
Figure 2-2. J-Flash SPI software in SEGGER	7
Figure 2-3. Open the J-Flash SPI software interface	7
Figure 2-4. Connect target SPI Flash	8
Figure 2-5. SPI Flash configuration interface	8
Figure 2-6. GD25Q16B parameter configuration	9
Figure 2-7. JLink successfully connected to SPI Flash	10
Figure 2-8. Open the downloaded binary file	10
Figure 2-9. File download to Flash successful prompt	11
Figure 2-10. Read data in Flash	11
Figure 3-1. New FLM project	12
Figure 3-2. Porting SPI driver and GD25qxx file	15
Figure 3-3. Compile and generate GD25Q16B.FLM file	16
Figure 3-4. Add GD25Qxx download algorithm to KEIL	18
Figure 3-5. Compile and download files in KEIL to SPI Flash	19



List of Table

Table 2-1. Jlink and SPI Flash hardware connection	6
Table 3-1. FlashPrg.c function interface	
Table 3-2. Implementation of FlashPrg.c Function Interface	
Table 3-3. FlashDevice structure realization	15
Table 3-4. Modify the pdsc file code	17
Table 4-1. Revision history	



1. Introduction

This application note uses the GD32F450i-EVAL board, the target chip is GD25Q16BS SPI nor flash, and the file is downloaded to the GD25Qxx SPI nor flash through the J-FLASH SPI host computer or modified KEIL download algorithm.



2. Use J-Flash SPI host computer to download files to SPI Nor Flash

2.1. Hardware connection

JLink supports the SPI protocol and connects the six wires: VTref, GND, TDI (MOSI), TMS (nCS), TCK (CLK), and TDO (MISO) in JLink to the pins of SPI Nor Flash. This application note uses the GD25Q16BS SPI nor flash chip in the GD32F450i-EVAL V1.1 development board. According to the schematic diagram of the development board and the JTAG pin diagram, as shown in *Figure 2-1. GD25Q16BS schematic diagram (left) and JTAG pin diagram (right)*, use the DuPont cable to connect the Jlink and the Flash hardware. The method is shown in *Table 2-1. Jlink and SPI Flash hardware connection*.



Figure 2-1. GD25Q16BS	S schematic diagram	(left) and J	JTAG pin diagran	ו (right)
-----------------------	---------------------	--------------	------------------	-----------

Table 2-1. Jlink and	d SPI Flash	hardware	connection
----------------------	-------------	----------	------------

JTAG pin number and name	Connect to the pins of GD25Q16BS
1(VTref)	Board VCC
5(TDI)	Board JP20 No. 3 pin (MOSI)
7(TMS)	Board PI8 pin(CS)
9(TCK)	Board JP13 No. 3 pin (SCK)
13(TDO)	Board PG12 pin(MISO)
4(GND)	Board GND

2.2. Jflash-SPI host computer configuration and download

First, double-click to open J-Flash SPI, as shown in <u>Figure 2-2. J-Flash SPI software in</u> <u>SEGGER</u>, and open the interface as shown in <u>Figure 2-3. Open the J-Flash SPI software</u> <u>interface</u>.



Figure 2-2. J-Flash SPI software in SEGGER



Figure 2-3. Open the J-Flash SPI software interface



Click "Target-->Connect", the result is shown in the red block diagram in <u>Figure 2-4. Connect</u> <u>target SPI Flash</u>. At this time, the SPI Flash Id has been successfully read, but the connection failed. Next, the Flash related parameters will be configured.



Figure 2-4. Connect target SPI Flash

SEGGER J-Flash SPI V6.50b - [C:\Users\haoran.chen\AppData\Roaming\SEGGER\Defaultjflash "] File Edit View Taraet Options Window Help	-	Х
Rene Value Connection USB (Device 0) Interface speed 12000 kHz Fisch memory Auto detection Plash dice 1 SEGGEER		
Application log started - J-Flash SFI W6 50b (J-Flash compiled Sep 6 2019 17:58:17) - JirkhAMR dll W6 50b (JLL compiled Sep 6 2019 17:57:38) Creating are project file (LWertharar achek)ApplatalRowsing(SEGGER/Default.jflash] Conserting - Conserting - Conserting - Conserting - Conserting - Target = 3.167V - Failed to auto-detect SFI Flash. - Failed to auto-detect SFI Flash.	(۲. ۲. «
Failed to connect		//

Click "Options-->Project settings", select FLASH, and uncheck automatically detect SPI flash. The reference interface is shown in *Figure 2-5. SPI Flash configuration interface*.

Figure 2-5. SPI Flash configuration interface

SEGGER J-Flash SPI V6.50b - [C:\Users\haor.	an.chen\AppData\Roaming\SEGGER\Default.jflash *] Help	-		×
Project - Def Project settings Name Value Global settings Connection USB [Device 0]	Alt-F7	×	1	
Interface speed 12000 kHz	Convert Lotter Elath Deschusion	^		
Flash nemory Auto detection Flash dice 1 Flash dice 1 Flash dice 1	Central Setur Plats Production Automatically detect Pl flash Single-Die flash device Flash ID Du00 Detect SPI flash NumPages Du00 Detect SPI flash VinteEnable Du00 NumPages Outor Detect SPI flash PageSize WinteEnable Du04 WinteStatus Du05 Control Instructions Status Register © Ready Bit Extenstruction Def3 Status Register Cater instruction Def3 Status Register EaseSector Du08 WintePage Du1 EaseBuk DuC7 ReadData Du03		2	2 2 2
	· · · · · · · · · · · · · · · · · · ·	(A)		
<				> .::
Ready				/

Refer to the GD25Q16B datasheet and fill in the relevant parameters such as the Flash page size, block size, read and write commands, and the specific configuration is shown in *Figure* <u>2-6. GD25Q16B parameter configuration</u>. After the configuration is complete, click "OK".



Figure 2-6. GD25Q16B parameter configuration



Click "Target-->Connect" again in the main interface, the result is shown in the red block diagram in *Figure 2-7. JLink successfully connected to SPI Flash*, and the relevant parameters are displayed on the left, and it prompts that JLink and Flash are successfully connected.



Figure 2-7. JLink successfully connected to SPI Flash



Click "File—>open data file" to open the binary file to be downloaded, as shown in *Figure 2-8. Open the downloaded binary file*.

SEGGER J-Flash SPI V6.50b - [C:\	\Users\haoran.chen\AppData\Roaming\SEGGER\Default.jflash *] -	×
File Edit View Target Options	Window Help	
Open data file Ctrl+O		
Merge data file		
Save data file Ctrl+S		
Save data file as	Enter start address X	
New contract		
New project	Start address Ox 🖸 OK	
Open project	Const	
Save project	Cancer	
Save project as	e	
Close project	· · · · · · · · · · · · · · · · · · ·	
Recent Files	Et\LED.bin	x
Recent Projects	Address: 0x0 x1 x2 x4	
Exit Alt+F4	0000 08 04 00 20 65 01 00 08 BB 01 00 08 B3 01 00 08 e	
WriteEnable 0x06	0010 B7 01 00 08 AD 01 00 08 39 02 00 08 00 00 009	
EraseBulk 0x60	0020 00 00 00 00 00 00 00 00 00 00 00 00	
ReadID 0x9F	0030 B1 01 00 08 00 00 00 E5 01 00 08 E9 01 00 08	
ReadStatus 0x05	0040 7F 01 00 08 7F 01 00 08 7F 01 00 08 7F 01 00 08 0	
WriteStatus UxU1	0050 7F 01 00 08 7F 01 00 08 7F 01 00 08 7F 01 00 08 0	
	0060 7F 01 00 08 7F 01 00 08 7F 01 00 08 7F 01 00 08 0	
	0070 7F 01 00 08 7F 01 00 08 7F 01 00 08 7F 01 00 08 0	
	0080 7F 01 00 08 7F 01 00 08 7F 01 00 08 7F 01 00 08 0	
LOG	9090 2F 01 00 08 00000000000000000000000000000	
- Verifying Flash Id = 0xC8, 0x40,	0 0000 2F 01 00 08 0 0 0 0	
Executing init sequence		
- Executed successfully		
- Reading affected sectors		
- Target memory read successful		
- Target programmed successfully -		
- 512 sectors, 1 range, 0x0 - 0x11	FFF U0F0 7F 01 00 08 7F 01 00 08 7F 01 00 08 00 00 00 00 00 0	
- Target memory read successfully.	0100 7F 01 00 08 00 00 00 7F 01 00 08 7F 01 00 08	
	0110 7F 01 00 08	-1
<	8108 00 A A A A A A A A A A A A A A A A A	-
Open an existing data file	Connected Speed: 8000	кΗ

Figure 2-8. Open the downloaded binary file

Click "Target-->Program", as shown in *Figure 2-9. File download to Flash successful prompt*, after the download is complete, it will prompt Target programmed successfully.



Figure 2-9. File download to Flash successful prompt

SEGGER J-Flash SPI V6.50b - [C:\Users\haoran.chen\AppData\Roaming\SEGGER\Default.jflash *] –										×														
File Edit View	ile Edit View Target Options Window Help																							
Droject - De	¢	Connect			J-I	Flas	h SP	I V6	.50b													Х		X
Froject - Del	_	Disconnect			1																			
Name	1				1																			
Connection	1	Test		>	H																			
Interface speed		Erase sectors		F3	1		IJ	Tar	get p	rogr	amn	ned s	succe	essfu	ılly -	Con	nplet	ed a	fter	0.080) sec			
Flash memoru	1	Erase chip		F4																			9	
Flash dice	<u> </u>		-																					
		Program		F5																		_		
Flash ID	_	Des servers Re Ma																		硽	÷			
FlashSize	1	Program & ve	rinj	/ FO																1410	~=		BB	
NumPages	1	Auto		F7													_						۰ه	
PageSize	4				7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵	۵۵	
SectorSize	+	Verify		F8	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵	۵۵	
NumAddrBytes	3	Read back		```	2F	Ø 1	ØЙ	Ø 8	7F	Ø1	ØЙ	Ø8	7F	Ø 1	ØЙ	Ø 8	2F	Ø1	ØЙ	Ø8	A	۵	AA	
4ByteAddrMode		Redd Ddek			200	Q1	00	60	יייי	61	66	60	70	Q1	00	60	70	61	60	60				
StatusisHeadyBit StatusPiPage	0.00		F	0070	nr.	01	00	00	71	01	00	00	71	01	00	00	7F	01	00	00	····	····		
Statusbichos	0x00		H	оно	71	01	00	08	7F	01	00	68	7F	01	90	68	γF	01	90	68	۵	۰	۰۵	
WriteEnable	0x06			00B0	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۰	۰۵	
WriteDisable	0x04			0000	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵	۵	
EraseBulk	0x60		IΓ	00D0	7F	01	00	08	7F	01	00	Ø8	7F	01	00	08	7F	01	00	08	۵	۵	۵۵	
ReadID	0x9F		۱ŀ	ØØFØ	7 F	Ø1	00	08	78	01	00	80	7 F	Ø 1	00	08	7 F	01	00	68	•	•	<u> </u>	
ReadStatus	0x05		۱ŀ	0010		01	00	00	-	01	00	00	-	01	00	20	00	00	00	00				
WriteStatus	0x01		Iŀ	0010	74	01	90	98	71	01	90	80	71	91	90	68	90	90	90	90	٥	•	o	
				0100	7F	01	00	68	00	00	00	00	7F	01	60	68	7F	01	00	68	۵		۰۵	
<		>	IL	0110	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵	۵	1
			Г	A10A	סר	R1	99	60	סר	R 1	00	00	סר	R1	99	60	70	R1	99	60	^	•		-

In order to verify whether the binary file is successfully downloaded to the Flash, through the "Target-->Read back-->Entire chip" operation, read the value of the address and compare it with the source file, as shown in *Figure 2-10. Read data in Flash*.

Figure 2-10. Read data in Flash

J	👷 SEGGER J-Flash SPI V6.50b - [C:\Users\haoran.chen\AppData\Roaming\SEGGER\Default.jflash *] — 🗆 🗙																											
File	e Edit	View	Tar	get	Options	5 V	Vind	low Hel	р																			
	Project	- Def	F	Con	nect				1																			×
	lame	- 1	1	Disc	onnect				10	_		-	1		us I													
C	onnection		1	Test				>	XU				141	<u>~</u>	<u>~</u>													
								-	0	1	2	3	4	5	6	7	8	9	A	B	С	D	E	F	ASCI	I		-
Ir	nterface sp	eed	1	Eras	e secto	rs		F3	08	04	30	20	65	01	00	08	BB	01	00	08	B 3	01	00	08		e		-
E	lash memo	m		Eras	e chip			F4	B7	01	00	08	AD	01	00	08	39	02	00	08	00	00	00	00		9		
F	lash dice	.,	1						00	00	00	00	00	00	00	00	00	00	00	00	E7	01	00	08				-
				Prog	Iram			F5	B1	01	00	08	00	00	00	00	E5	01	00	08	E9	01	00	08				1
I F	lash ID		1	Prog	ram &	Veri	ify	F6	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۰۵		
N	lumPages		1	Auto			1	F7	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۰۵		
P	ageSize		-	Auto					7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵۵		
S	ectorSize			Veri	fy			F8	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۰۵		
N	lumAddrBy	tes		Read	l back			>		Ent	ire cł	nin		٦	00	08	7F	01	00	08	7F	01	00	08	۵	۵۵		
4	byteAddrM tatusIsBea	aduBit	I STILL					0070		0			_		00	08	7F	01	00	08	7F	01	00	08	۵	۵۵		
Š	tatusBitPo	s	0x00					ØØAØ		Kar	nge	~~~		_	00	08	7F	01	00	08	7F	01	00	08	۵	۵۵		
								ØØRØ	2F	Ø1	ØØ	08	7F	Ø1	ØЙ	Ø8	7 F	Ø1	ØЙ	08	7 F	Ø1	00	08	۵			
W	/riteEnable	8	0x06			- 11		0020	2F	Ø1	ØЙ	08	7F	Ø1	ØЙ	Ø8	7F	Ø1	ØЙ	08	7F	Ø1	ØЙ	08	<u>_</u>	 		
E	raseBulk	e	0x04			- 11		ØØDØ	28	Ø1	00	08	7F	Ø1	00	08	7F	Ø1	00	08	7F	Ø1	00	08	~	~ ~	~	
R	leadID		0x9F				F	OOFO	28	Ø1	00	00	78	Ø1	00	00	78	Ø1	00	60	78	Ø1	00	60	~	~ ~	····-	
R	leadStatus	:	0x05				H	OOLO	28	01	00	60	72	01	00	60	71	01	00	60	00	00	00	00		~ ~		
W	/riteStatus		0x01			- 11	\vdash	0100	71	61	00	60	00	00	00	00	71	01	00	60	70	60 61	00	60	· · · · ·			
							H	0110	71	01	00	00	99	00	00	60	71	01	00	80	71	01	00	00	····			
<					2			0120	71	01 01	88	80	ייי סרי	01 01	99	80	71	01 01	88	80	ייי סרי	01 01	88	80	·····	·····	····a···	•
	-						_																					
5	LOG																											8
	Verifyin	ng Fla	ish Id	= 0x0	8, 0x40	l, Ox	15.	0.K.																				^
Ex	ecuting :	ea suc init s	equen	шту се																								
- P.,	Execute	d succ	essfu	lly 102 Lu																								
'r	Reading	g targ affec	ted s	≇9∠ Dy ectors	i	ranş	ge/																					
	- Targ	et mem	ory r	ead su	ccessfu	illy.	(4)	096 bytes	., 1 r	ange)																	
Re	- Target programmed successfully - Completed after 0.080 sec Reading entire flash chip																											
-	- 512 sectors, 1 range, 0x0 - 0x1FFFFF																											
-	- iarget memory read successfully. (209/152 bytes, 1 range) - Completed after 2. (/9 sec																											
<																												>
Rea	id the en	tire fla	ash ch	ip																			(Conn	ected	Spe	ed: 8000	kH /



3. Use KEIL to download files to SPI Nor Flash

3.1. New FLM project

Enter the drive letter where KEIL is installed, copy the Keil\ARM\Flash_Template project to the Test folder of Disk E (the folder location can be modified as needed), double-click to open the "NewDevice.uvprojx" project, compile the project, the project will report an error "FlashDev. c(25): error: #5: cannot open source input file "..\FlashOS.H": No such file or directory", enter againFind the "FlashOS.h" file in the Keil\ARM\Flash directory, copy it to the "E:\Test_Template" directory, and change the #include "../FlashOS.H" in FlashDev.c and FlashPrg.c to #include "FlashOS.H", compile the project again, there is no error in the project, and generate NewDevicec.FLM. Related projects and compilation are shown in *Figure 3-1. New FLM project.*

Figure 3-1. New FLM project



3.2. Porting SPI Flash driver code

Open the FlashPrg.c file, which mainly contains seven function interfaces, as shown in <u>Table</u> <u>3-1. FlashPrg.c function interface</u>.

Table 3-1	. FlashPrg.c	function	interface
-----------	--------------	----------	-----------

/* Flash Programming Functions (Called by FlashOS) */					
extern	int	Init	(unsigned long adr,	/* Initialize Flash */	
unsigned long clk,					



AN017 Use Jlink to download files to SPI Nor Flash

unsigned long fnc);					
extern	int	UnInit	(unsigned long fnc);	/* De-initialize Flash */	
extern	int	BlankCheck	(unsigned long adr,	/* Blank Check */	
			unsigned long sz,		
unsigned char pat);					
extern	int	EraseChip	(void);	/* Erase complete Device */	
extern	int	EraseSector	(unsigned long adr);	/* Erase Sector Function */	
extern	int	ProgramPag	ge (unsigned long adr,	/* Program Page Function */	
			unsigned long sz,		
unsigned char *buf);					
extern unsigned	long	Verify	unsigned long adr,	/* Verify Function */	
			unsigned long sz,		
			unsigned char *buf);		

It mainly implements Init, EraseChip, EraseSector, ProgramPage and Verify function interfaces. The function interface implementation is shown in <u>Table 3-2. Implementation of</u> <u>FlashPrg.c Function Interface</u>.

 Table 3-2. Implementation of FlashPrg.c Function Interface

```
uint32_t base_adr;
   Initialize Flash Programming Functions
 *
      Parameter:
                       adr: Device Base Address
                        clk: Clock Frequency (Hz)
                        fnc: Function Code (1 - Erase, 2 - Program, 3 - Verify)
                       0 - OK, 1 - Failed
      Return Value:
 */
int Init (unsigned long adr, unsigned long clk, unsigned long fnc) {
 /* Add your Code */
  spi_flash_init();
  base_adr = adr;
  return (0);
                                                 /* Finished without Errors */
   Erase complete Flash Memory
      Return Value: 0 - OK, 1 - Failed
 */
int EraseChip (void) {
 /* Add your Code */
 spi_flash_bulk_erase();
  return (0);
                                                  /* Finished without Errors */
```



* Erase Sector in Flash Memory * Parameter: adr: Sector Address * Return Value: 0 - OK, 1 - Failed */ int EraseSector (unsigned long adr) { /* Add your Code */ spi_flash_sector_erase(adr); return (0); /* Finished without Errors */ * Program Page in Flash Memory * Parameter: adr: Page Start Address SZ: Page Size buf: Page Data 0 - OK, 1 - Failed Return Value: */ int ProgramPage (unsigned long adr, unsigned long sz, unsigned char *buf) { /* Add your Code */ spi_flash_page_write(buf,adr,sz); return (0); /* Finished without Errors */ unsigned long Verify (unsigned long adr, unsigned long sz, unsigned char *buf) uint8_t readbuf[256]; uint32_t len; uint32_t count = 0; uint32_t readcount = 0; uint32_t readaddrs = 0; if((sz%256)==0) { readcount = sz/256; }else { readcount = sz/256 + 1; } readaddrs = (adr - base_adr); for(count=0;count<readcount;count++)</pre> {



spi_flash_buffer_read(readbuf,(readaddrs+count*256),256);
for(len=0;len<256;len++)
{
 if(buf[len+count*256] != readbuf[len])
 {
 return count*256 + adr + len;
 }
}
return adr+sz;</pre>

The related SPI driver is added to the KEIL project according to GD32F4xx_Firmware_Library and GD25qxx.c, and the added files are shown in *Figure 3-2. Porting SPI driver and* <u>GD25qxx file</u>.

Figure 3-2. Porting SPI driver and GD25qxx file



3.3. Modify FlashDevice structure

Open the FlashDev.c file and modify the relevant content in the FlashDevice structure. The modified code is shown in <u>Table 3-3. FlashDevice structure realization</u>.

 Table 3-3. FlashDevice structure realization

struct FlashDevice const FlashDevice = {				
FLASH_DRV_VERS,	/* Driver Version, do not modify!			
"GD25qxx",	/* Device Name */			
EXTSPI,	/* Device Type */			
0x00000000,	/* Device Start Address */			
0x00200000,	/* Device Size in Bytes (2M) */			
256,	/* Programming Page Size */			
0,	/* Reserved, must be 0 */			



0xFF,	/* Initial Content of Erased Memory */
100,	/* Program Page Timeout 100 mSec */
3000,	/* Erase Sector Timeout 3000 mSec */
/* Specify Size and Address of Sec	tors
0x001000, 0x000000,	/* Sector Size 4kB (4096 Sectors) */
// 0x010000, 0x010000,	/* Sector Size 64kB (2 Sectors) */
// 0x002000, 0x030000,	/* Sector Size 8kB (8 Sectors) */
SECTOR_END	
};	

3.4. Compile and generate FLM file

Open the magic wand, enter the Output page, modify the Name of Executable to GD25Q16B, compile the project, and generate the GD25Q16B.FLM file. As shown in *Figure 3-3. Compile and generate GD25Q16B.FLM file*.



Figure 3-3. Compile and generate GD25Q16B.FLM file



3.5. Add algorithm file to KEIL project

Copy the compiled GD25Q16B.FLM to the KEIL installation directory, D:\Keil_527\ARM\PACK\GigaDevice\GD32F4xx_DFP\2.0.0\Flash, and then return to the upper-level directory to open the GigaDevice.GD32F4xx_DFP.pdsc file and modify its attributes To read and write, find GD32F450IK, add the code as shown in red in <u>Table 3-4.</u> <u>Modify the pdsc file code</u>.

Table 3-4. Modify the pdsc file code

In the KEIL project, open the magic wand Utilities page setting, and add the GD25qxx algorithm, as shown in *Figure 3-4. Add GD25Qxx download algorithm to KEIL*.



Figure	3-4.	Add	GD25Qxx	download	algorithm	to KEIL
riguic	J- 1 .	Auu	ODZJĘXX	aowinioau	argoritinn	

options for larger GD524501_EVAL	
vice Target Output Listing User C/C++	Asm Linker Debug Utilities
Configure Flash Menu Command	
Use Target Driver for Flash Programming	2 Use Debug Driver
Use Debug Driver	Settings Vodate Target before Debugging
Init File:	Edt
C	
Use External Tool for Hash Programming	
Command:	
Arguments.	
Hun Independent	
Configure Image File Processing (FCARM):	
Output File:	Add Output File to Group:
	Application
Image Files Root Folder:	Generate Listing
OK C	Cancel Defaults Help
Download Function	RAM for Algorithm
Download Function LOND ← Erase Full Chip ☞ Program ← Erase Sectors ☞ Verify	RAM for Algorithm Start: 0x20000000 Size: 0x3000
Download Function C Erase Full Chip IF Program C Erase Sectors IF Verify C Do not Erase IF Reset and Run	RAM for Algorithm Start: 0x20000000 Size: 0x3000
Download Function C Erase Full Chip IF Program C Erase Sectors IF Verify C Do not Erase IF Reset and Run Programming Algorithm	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Flash Programming Algorithm Peterstein Page Tax Date
Download Function C Erase Full Chip IP Program C Erase Sectors IP Verify C Do not Erase I Reset and Run Programming Algorithm Description Device Size Dev	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm Size: 0x3000 Size: 0x3000 Add Eash Programming Algorithm Size: 0x3000 Size: 0x3000 Add Flash Programming Algorithm Size: Size: 0x3000 Size: 0x3000 Add Size: Size: Size: Size: Size: Size: Size: Add Size: Size:<
Download Function	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Flack Programming Algorithm Size: 0x3000 Size: 0x3000 Add Flack Programming Algorithm Size: Size: </td
Download Function C Erase Full Chip Program F Erase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev 3	RAM for Algorithm Size: 0x3000 Size: 0x3000 Size: 0x3000 Add Eash Programming Algorithm Colspan="2">Colspan="2" Colspan="2" <th colspan="2</td>
Download Function C Erase Full Chip F Program F Erase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev 3 Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Flash Proce first 0x6000000 Size: 0x3000 Add Flash Proce first 0x60000000 Size: 0x30000 Add Flash Proce first 0x60000000 Size: 0x30000 Colore But Size 0x600000000000000000000000000000000000
Download Function C Erase Full Chip F Program F Erase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev 3 Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm Control Fash The Control Fash Plane Control Fas
Download Function C Erase Full Chip F Program F Erase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev 3 Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm Total Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm Total Start: 0x20000000 Size: 0x3000 Colord Start Color Start Start Colord Start Color Start Start Colord Start Color Start Start Start: 0x20000 Start Start Start: 0x2000 Start Start Start: 0x2000 Start Start Start: 0x2000 Start Start Start: 0x2000 Start Start
Download Function C Erase Full Chip F Program F Erase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm: Size: 0x3000 Size: Size: Control Finite Size: Size: Size: Size:
Download Function C Erase Full Chip Program Frase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev 3 Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm COUCH and the second sec
Download Function C Erase Full Chip Program Erase Sectors Verify C Do not Erase Reset and Run Programming Algorithm Description Device Size Dev 3 Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm Totants 0x20000000 Size: 0x3000 Add Eash Programming Algorithm Totants 0x100000000000000000000000000000000000
Download Function C Erase Full Chip Program Frase Sectors Verify Do not Erase Reset and Run Programming Algorithm Description Device Size Dev Add	RAM for Algorithm Start: 0x20000000 Size: 0x3000 Add Eash Programming Algorithm

3.6. Compile and download

Compile the project in KEIL, generate the .axf file, click the Download button to download the file, as shown in *Figure 3-5. Compile and download files in KEIL to SPI Flash*, indicating that the download is successful.



Figure 3-5. Compile and download files in KEIL to SPI Flash



3.7. Testing and verification

In order to determine whether the file is successfully downloaded to the GD25Q16BS Flash, refer to <u>Jflash-SPI host computer configuration and download</u>, read the data in Flash through the J-Flash SPI host computer, compare whether the downloaded file and the read file are the same, and perform a verification test.



4. Revision history

Table 4-1. Revision history

Revision No.	Description	Date
1.0	Initial Release	Apr.30, 2021



Important Notice

This document is the property of GigaDevice Semiconductor Inc. and its subsidiaries (the "Company"). This document, including any product of the Company described in this document (the "Product"), is owned by the Company under the intellectual property laws and treaties of the People's Republic of China and other jurisdictions worldwide. The Company reserves all rights under such laws and treaties and does not grant any license under its patents, copyrights, trademarks, or other intellectual property rights. The names and brands of third party referred thereto (if any) are the property of their respective owner and referred to for identification purposes only.

The Company makes no warranty of any kind, express or implied, with regard to this document or any Product, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The Company does not assume any liability arising out of the application or use of any Product described in this document. Any information provided in this document is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Except for customized products which has been expressly identified in the applicable agreement, the Products are designed, developed, and/or manufactured for ordinary business, industrial, personal, and/or household applications only. The Products are not designed, intended, or authorized for use as components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, atomic energy control instruments, combustion control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or Product could cause personal injury, death, property or environmental damage ("Unintended Uses"). Customers shall take any and all actions to ensure using and selling the Products in accordance with the applicable laws and regulations. The Company is not liable, in whole or in part, and customers shall and hereby do release the Company as well as it's suppliers and/or distributors from any claim, damage, or other liability arising from or related to all Unintended Uses of the Products. Customers shall indemnify and hold the Company as well as it's suppliers and/or distributors harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of the Products.

Information in this document is provided solely in connection with the Products. The Company reserves the right to make changes, corrections, modifications or improvements to this document and Products and services described herein at any time, without notice.

© 2021 GigaDevice - All rights reserved