GigaDevice Semiconductor Inc.

Arm® Cortex®-M3/4/23/33 32-bit MCU

应用笔记 AN017



目录

E	录		2
表	表索引.		3
图	国索引.		4
1	.简介	·	5
2	. 使用	J-Flash SPI 上位机下载文件到 SPI Nor Flash	6
	2.1.	硬件连接	6
	2.2.	Jflash-SPI 上位机配置与下载	6
3	. 使用	「KEIL 下载文件到 SPI Nor Flash1	2
	3.1.	新建 FLM 工程1	2
	3.2.	移植 SPI Flash 驱动代码 1	2
	3.3.	修改 FlashDevice 结构体1	5
	3.4.	编译生成 FLM 文件1	6
	3.5.	添加算法文件到 KEIL 工程中1	6
	3.6.	编译和下载1	8
	3.7.	测试和验证1	9
4	. 历史	版本2	20



表索引

表 2-1. Jlink 与 SPI Flash 硬件连接	6
表 3-1. FlashPrg.c 函数接口	12
表 3-2. FlashPrg.c 函数接口的实现	
表 3-3. FlashDevice 结构体实现	
表 3-4. 修改 pdsc 文件代码	17
表 4-1. 历史版本	20



图索引

图 2-1. GD25Q16BS 原理图(左)与 JTAG 引脚图(右)	6
图 2-2. SEGGER 中 J-Flash SPI 软件	7
图 2-3. 打开 J-Flash SPI 软件界面	7
图 2-4. 连接目标 SPI Flash	8
图 2-5. SPI Flash 配置界面	8
图 2-6. GD25Q16B 参数配置	9
图 2-7. JLink 成功连接 SPI Flash	9
图 2-8. 打开下载的二进制文件	10
图 2-9. 文件下载到 Flash 成功提示	10
图 2-10. 读取 Flash 中的数据	11
图 3-1. 新建 FLM 工程	
图 3-2 移植 SPI 驱动和 GD25qxx 文件	15
图 3-3. 编译生成 GD25Q16B.FLM 文件	
图 3-4. KEIL 中添加 GD25Qxx 下载算法	
图 3-5. KEIL 中编译和下载文件到 SPI Flash	19



1. 简介

本应用笔记采用 GD32F450i-EVAL 开发板,目标芯片为 GD25Q16BS SPI nor flash 芯片,通过 J-FLASH SPI 上位机或者修改 KEIL 下载算法,将文件到 GD25Qxx SPI nor flash 中。



2. 使用 J-Flash SPI 上位机下载文件到 SPI Nor Flash

2.1. 硬件连接

JLink 支持 SPI 协议,将 JLink 中的六根线 VTref、GND、TDI (MOSI)、TMS (nCS)、TCK (CLK)、TDO (MISO)对应连接到 SPI Nor Flash 的引脚上。本应用笔记采用 GD32F450i-EVAL V1.1 开发板中的 GD25Q16BS SPI nor flash 芯片,根据开发板原理图和 JTAG 引脚图,如 <u>*图* 2-1. GD25Q16BS 原理图 (左)</u>与 JTAG 引脚图(右)</u>所示,使用杜邦线将 Jlink 与 Flash 硬件连接,连接方法参考<u>表 2-1. Jlink 与 SPI Flash 硬件连接</u>。

图 2-1. GD25Q16BS 原理图(左)与 JTAG 引脚图(右)



表 2-1. Jlink 与 SPI Flash 硬件连接

JTAG 引脚编号及名称	连接到 GD25Q16BS 的引脚
1(VTref)	开发板 VCC
5(TDI)	开发板 JP20 3 号引脚(MOSI)
7(TMS)	开发板 PI8 引脚(CS)
9(TCK)	开发板 JP133号引脚(SCK)
13(TDO)	开发板 PG12 引脚(MISO)
4(GND)	开发板 GDN

2.2. Jflash-SPI 上位机配置与下载

首先单击打开 J-Flash SPI,如图 2-2 所示,打开界面见<u>图 2-2. SEGGER 中 J-Flash SPI 软</u> <u>件</u>。



图 2-2. SEGGER 中 J-Flash SPI 软件



图 2-3. 打开 J-Flash SPI 软件界面

SEGGER J-Flash SPI V6.50b - [C:\Users\haoran.chen\AppData\Roaming\SEGGER\Default;flash *] - C X												
File Edit View Target Options \	Vindow Help											
Project - Def Value Connection USB [Device 0] Interface speed 12000 kHz Flash memory Auto detection Flash dice	SEGGER											
Application log started Application log started - Think STI W. 50b (JF) And compile - Think MR 100 50b (ILL compiled) Creating new project file [C:Wsersh - New project created successfully	nd Sep & 2019 17:58:17) Sep & 2019 17:57:35) soran chen\AppData\Roaning\SEGGER\Default.jflash]			× <								
Ready				11.								

点击"Target-->Connect",可以看到<u>图 2-3. 打开 J-Flash SPI 软件界面</u>中红色框图,此时已 成功读出 SPI Flash Id,但是连接失败,接下来将配置 Flash 相关参数。



图 2-4. 连接目标 SPI Flash

🛃 SEGGER J-Flash SPI V6.50b - [C:\Users\haoran.chen\AppData\Roaming\SEGGER\Default.jflash *]	-		×
File Edit View Target Options Window Help			
Project - Def., Value Value Connection USB (Device 0) Interface speed 12000 MHz Plash memoy Auto detection Plash dee 1			
		•	23
Application log started - J-Falah STW 650b (J-Flach compiled Sep 6 2019 17:58:17) - JLinhAMM dll W6.50b (DLL compiled Sep 6 2019 17:57:35) Crating mey project file (C. Wayrihavan AnchayBplathMosaing/SEGGER/Default.jflach]			^
Conneting - Conneting via USB to J-Link device 0 - Virget = 3.167V - Read SFT Jack Id = 0.628 40 15 - ERGUR: Failed to auto-detect SFT Flach. - Failed to auto-detect SFT Flach.			ý
			> .::
Failed to connect			//

点击"Options-->Project settings",选择 FLASH,将 Automatically detect SPI flash 取消勾选,参考界面<u>图 2-5. SPI Flash 配置界面</u>。

图 2-5. SPI Flash 配置界面

Project - Def Project settings Alt-F7 Global settings Connection USB [Device 0] Interface speed 12000 kHz Flash doe 1 Project settings Flash Doo Doo Doo Doeted SP Hash Flash doe 1 Device 10 Project settings Flash Doo Doo Doeted SP Hash Strate Doeted SP Hash	SEGGER J-Flash SPI V6.50b - [C:\Users\ File Edit View Target Options Wind	aoran.chen\AppData\Roaming\SEGGER\Default.jflash *] ow Help	-	×
Flash memory Auto detection Flash memory 1 Flash dice 1 Series Setting: Single-De flash device Hash ID Ext and active device NumPage Rott PageSize Control Instruction Control Instruction ViteDiable 0x06 Page Size Instruction 0x67 Page Dott Control Instruction Page Instruction 0x67 Page Instruction	Project - Def Project set Name Value Global sett Connection US8 [Device 0] Interface speed 12000 kHz	Ings Alt-F7 Ings Project settings ? General Setur Flash Production	×	
	Flash memory Auto detection Flash dice 1 Flash dice 1 Auto detection Flash dice 1 Auto detection Flash memory Constrained and the state of th	Automotically detect SPI Bash Single-Die Bash device Exercise Settings Flash ID 0:000 Device SPI Bash NumPages 0:000 PageSize 0:01 Variate Table 0:000 Variate Table 0:000 PageSize 0:001 PageSize 0:001 PageSize 0:005 Ready Bit Bit Post PageSize 0:008 VirtePage 0:02 ErareBuk 0:07 ReadData 0:03		
	<			×

参考 GD25Q16B datasheet,填写 Flash 页大小、块大小、读写命令等相关参数,具体配置参 考 <u>图 2-6. GD25Q16B 参数配置</u>,配置完成后点击确定。



图 2-6. GD25Q16B 参数配置

Project settings	?	×
General Setup Flash Production		
General Settings		
Flash ID 0xC8 0x40 0x15 Detect SPI flash		
NumPages 0x2000 PageSize 0x100		
NumAddrBytes 0x03 SectorSize 0x1000		
Control Instructions		
WriteEnable 0x06 ReadStatus 0x05 ReadID 0x3F WriteDisable 0x04 WriteStatus 0x01 0x3F		
Dedicated 4-byte addr. mode Status Register Enter instruction 0x87 Exit instruction 0xE9 Busy Bit		
Programming Instructions		
EraseSector 0x20 WritePage 0x02		
EraseBulk 0x60 ReadData 0x03		
确定 取消	应用(A)

在主界面中再次点击"Target-->Connect",可以看到<u>**82-7**. JLink</u>成功连接 SPI Flash</u>所示 红框,此时在左边显示相关参数,并且提示 JLink 与 Flash 连接成功。

图 2-7. JLink 成功连接 SPI Flash





点击 "File—>open data file",打开需要下载的二进制文件,如<u>图 2-8. 打开下载的二进制文件</u> 所示。

J SPI	SEGGER J-F	lash SPI V6.	.50b - [C:\Us	ers\haoran.cl	nen\A	ppD	ata∖F	Roam	ning\	SEG	GER	Defa	ult.jf	lash	*]								_		×
File	Edit Viev	w Target	Options V	Vindow Hel	р																				
	Open data	file	Ctrl+O																						
	Merge dat	a file																							
	Save data	file	Ctrl+S	E.c.			1		_	_	_	_	_	_	~										
	Save data	file as		Ente	r star	τado	aress	5							^										
	New proje	ct		0			0.1				_			,	1										
	Open proj	ect		Start	addre	SS	UX	<u>u</u>					10	< <u> </u>											
	Save proje	ct											Can	cel											
	Save proje	ct as											_												
	Close proj	ect	4																						
	Recent File	s	>	🛃 E:\ LED.I	oin																				×
	Recent Pro	jects	>	Address:	0x0	_	_		x <u>1</u>	x <u>2</u>	×4														
	m.da		Alt CA	Address	0	1	2	3	4	5	6	7	8	9	A	B	С	D	E	F	ASC	:11			
1.1.1.1	Exit	0.00	AIL+F4	0000	08	04	00	20	65	01	00	08	BB	01	00	08	B3	01	00	08		е.			
Wri	iteDisable	0x06		0010	B7	01	00	08	AD	01	00	08	39	02	00	08	00	00	00	00		• • •	9		-
Era	iseBulk	0x60		0020	00	00	00	00	00	00	00	00	00	00	00	00	E7	01	00	08	• • •	•••		• • • • • •	-
Re	adi.v adStatus	0x9F		0030	81	01	90	80	90	00	00	99	E5	01	00	80	EA	01	90	80					•
Wri	iteStatus	0x01		0040	75	01	90	00	יר סר	01 01	99	00	יוי סרי	01 01	00	80	75	01 01	99	00	····				•
				0050	7F	Ø1	ØЙ	08	71	Ø1	00 00	00 08	7F	Ø1	00 00	00	7F	Ø1	00 00	08	A	-0-			
<			>	0070	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵				
				0080	7 F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	. ۵.			
SPI	LOG			0090	7 F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	.۵.			
- v - c	Connected s	lash Id = 0) uccessfullv	xC8, 0x40, 0	00A0	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	.۵.			
Exec	uting init	sequence		00B0	7 F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	.۵.			
Pros	ramming ta	ccessfully rget (1492 b	bvtes, 1 ran	0000	7 F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	. ۵.			
- F	Reading aff	ected sector	rs	00D0	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	.۵.			
- T	- Target mo Carget progr	emory read : rammed succe	successfully essfully - C	00E0	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	.۵.			
Read	ling entire	flash chip		00F0	7F	01	00	08	7F	01	00	08	7F	01	00	08	00	00	00	00	۵	.۵.			
- E	olZ sectors, Carget memor	. 1 range, (rv read suc	uxu - Ox1FFF cessfully (0100	7F	01	00	08	00	00	00	00	7F	01	00	08	7F	01	00	08	۵				
	a Sec mentor	.,	·····	0110	7 F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	. ۵.			
<				01 D0	90	61	88	00	סרי	R1	66	60	70	R 1	66	60	70	61	88	60	•	^	^		-
Oper	n an existing	g data file		,										-					C	onn	ected		Spe	ed: 800	0 kH)

图 2-8. 打开下载的二进制文件

点击 "Target-->Program",如*图* 2-9. 文件下载到 Flash 成功提示所示,下载完成后,提示 Target programmed successfully。

图 2-9. 文件下载到 Flash 成功提示

J SPI	SEGGER J-Fla	ash SP	I V6.50b - [C:\	Use	ers\haoran.ch	en∖A	\pp[Data∖	Roar	ning	\SEG	GER	Defa	ault.jf	flash	*]							-		1	×
File	Edit View	Tar	get Options	W	/indow Help	p																				
J	Project - De	-	Connect			J-1	Flas	h SP	I V6	.50b	,						-						×			×
N	ame	1	Disconnect																							_
Co	onnection	1	Test		>	Ĥ																				
In	terface speed		Erase sector	s	F3			I)	Tar	get p	rogr	amn	ned s	succe	essfu	ılly -	Con	nplet	ed a	fter	0.080	0 sec				
FL	ash memory		Erase chip		F4																			9		
F.	ash dice		Program		F5																		-			
FI.	ash ID ashSize		Program &	/eri	fy F6	1															确	定		هه	•	
N	umPages	j –	Auto		F7									-										هه	•	
Pa St	ageSize actorSize	1	Varify		F8	7F	01	90	80	7F	Ø1	00	80	7F	01	90	08 08	7F	01 01	00	80	<u>م</u>	<u>م.</u> .		·	
N	umAddrBytes	1	Peed bask		10	25	01 01	00	08 08	75	01 01	00 00	68 08	75	01 01	00 00	08 08	7F	Ø1	00 00	08 08	۵ ۸	••••		·	
48	ByteAddrMode	1	Read back			-7F	61	66	00	78	01	00	60	78	61	00	88	75	61	00	60	· · · ·	~			
St	atusBitPos	0x00			0010	25	Q1	00	00	75	Ø1	00	00	7F	Q1	00	08	7F	01	00	60	^	~	· · · ·		
					ØØRØ	2F	Ø1	00	00	7F	Ø1	ØЙ	08	7F	Ø1	00	08	7F	Ø1	ØЙ	00	^	~	~ ~ ~		
W	riteEnable	0x06			0000	20	01	00	60	70	01	00	60	70	01	99	60	70	61	60	60		~			
W	riteDisable	0x04			0000	n n	01	00	00	71	01	00	00	nr nr	01	00	00	71	01	00	00	····	••••		••••	
E E	aseBulk.	0.00			0000	71	01	99	68	7F	01	00	68	7F	01	99	68	γF	01	99	68	۵	<u>م</u>			
	and Status	0x3F			00E0	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵		····	
	riteStatus	0x00			00F0	? F	01	00	08	7F	01	00	08	7F	01	00	08	00	00	00	00	۵	۵			
		001			0100	7F	01	00	08	00	00	00	00	7F	01	00	08	7F	01	00	08	۵		هه	·	
1	_				0110	7F	01	00	08	7F	01	00	08	7F	01	00	08	7F	01	00	08	۵	۵		· · · ·	
Ľ					B1 2B	סר	R1	99	60	90	R1	90	60	70	R1	99	60	70	R1	66	60	•	•			-

为了验证是否成功下载二进制文件到 Flash 中,可以通过"Target-->Read back-->Entire chip" 操作,读出所在地址的值,与源文件进行比对,如<u>图2-10. 读取 Flash 中的数据</u>所示。



图 2-10. 读取 Flash 中的数据

J	SEGGER J-Flash SPI V6.50b - [C:\Users\haoran.chen\AppData\Roaming\SEGGER\Default.jflash *] — 🗆 🗙																												
Fil	e Edit	View	Targ	jet (Optior	ns N	Nind	low He	p																				
	Project	t - Def		Conr	nect				I																			. •	
Г	Name	- (1	Disc	onnec	t			NO.	_		-	v1	v2	v4														
	Connection		i i	Test				>	a	4	2	2	4	 	<u></u>	7	0	a	0	р	C	D	F	P	00	011			
h	nterface sp	eed	1	Eras	e sect	ors		F3	08	04	30	20	65	01	00	08	BB	9 01	00	08	B3	01	00	08	H0 	. e			
F	lash memo	лy	1	Eras	e chip			F4	B7	Ø1	00	Ø8	AD	Ø1	00	Ø8	39	Ø2	00	Ø8	00	00	00	00	••	• • •	9		
F	lash dice			Prog	ram			F5	B1	00 01	00 00	00 08	00 00	00 00	00 00	00 00	00 E5	00 01	00 00	00 08	E7 E9	01 01	00 00	08 08					·: _
F	lash ID lashSize		1	Prog	iram 8	k Ver	ify	F6	7F	01	00	08	7F	01	00	08	7 F	01	00	08	7 F	01	00	08	۵.	۵	۵۵	.د	
I P	lumPages PageSize		1	Auto				F7	- 7F	01 01	00 00	08 08	7F 7F	01 01	00 00	08 08	7F 7F	01 01	00 00	Ø8 Ø8	7F 7F	01 01	00 00	08 08	۵.	ەە	ەە		
	ectorSize	iteo	_	Verif	ý		_	F8	7 F	01	00	08	7F	01	00	08	7F	01	00	08	7 F	01	00	08	۵.				
į	ByteAddrM	fode		Read	d back	:		>		Enti	ire ch	nip			00 00	08 08	7F 7F	01 01	00 00	Ø8 Ø8	7F	01 01	00 00	Ø8 Ø8	۵.	•••	ه	····	
9	itatusBitPo	S S	0x00				E	00A0		Rar	ige				00	Ø8	7F	01	00	Ø8	7F	01	00	08	۵.	۵	a		
	VriteEnable	в	0x06					00B0	7F	01 01	00 00	Ø8	7F	01 01	00 00	Ø8	7F	01 01	00 00	Ø8	7F	01 01	00	Ø8	۵.	ه	ە	<u>م</u>	
E	VriteDisabl TraseBulk	e	0x04 0x60					00D0	7F	01	00	08	7F	01	00	08	7F 7F	01	00	08	7F	01	00	08	۵.	۵۵ م۵	۵۰۰۰۵ ۵		
F	ReadID ReadStatus	,	0x9F					00E0	? F	01	00	08	7 F	01	00	08	7 F	01	00	08	7 F	01	00	08	۵.	۵	ەە	.د	
lli	VriteStatus		0x01				H	00F0	7F	01 01	00 00	Ø8 09	7F	01 00	00 00	Ø8 00	7F	01 01	00 00	Ø8 69	00 75	00 01	00 00	00 09	۵.	۵	ە		
Ι.	,							0110	7F	01	00	08	99 7F	01	00	08 08	7F 7F	01	00	08	7F	01	00	08	Δ.	 	۵۰۰۰۵ ۵		
Ľ						<u>_</u>		01 7 D	30	R 1	88	00	סרי	R 1	99	66	סרי	R 1	66	60	70	R 1	66	60	•	^	•	•	-
	LOG																												X
En	 Verifying Flash Id = 0xC8, 0x40, 0x150.K. Connected successfully Executing init sequence Executed successfully target (1422 bytes, 1 range) Reading affected sectors Target programmed successfully - Completed after 0.080 sec Reading ref. Flash oh: Flast oncessfully Completed after 0.080 sec 																												
<	<u>ا</u> ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا ا																												
Rea	ad the en	tire fla	ash chi	р																				Conn	necte	ed	Sp	eed: 80	00 kH



3. 使用 KEIL 下载文件到 SPI Nor Flash

3.1. 新建 FLM 工程

进入安装 KEIL 的盘符,将 Keil\ARM\Flash_Template 工程拷贝到 E 盘 Test 文件夹中(可自 己设定),双击打开 "NewDevice.uvprojx"工程,编译该工程,工程会报错 "FlashDev.c(25): error: #5: cannot open source input file "..\FlashOS.H": No such file or directory",再次进入 Keil\ARM\Flash 目录下找到 "FlashOS.h"文件,拷贝到 "E:\Test_Template" 目录下,将 FlashDev.c 和 FlashPrg.c 中的 #include "../FlashOS.H" 修改为 #include "FlashOS.H",再次 编译工程,工程没有错误,并生成 NewDevicec.FLM。相关工程及编译如<u>图 3-1. 新建 FLM 工</u> 推</u>所示。

图 3-1. 新建 FLM 工程



3.2. 移植 SPI Flash 驱动代码

打开 FlashPrg.c 文件,该文件主要包含七个函数接口,如表 3-1. FlashPrg.c 函数接口所示

表 3-1. Flas	hPrg.c	函数接			
/* Flash Prog	ramming	g Functio	ns (Called by FlashOS) */		
extern	int	Init	(unsigned long adr,	/* Initialize Flash */	
			unsigned long clk,		
			unsigned long fnc);		



AN017 基于 Jlink 烧录文件到 SPI Nor Flash 的方法

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)	,

这里主要实现 Init、EraseChip、EraseSector、ProgramPage 和 Verify 函数接口,函数接口实现如*表 3-2. FlashPrg.c 函数接口的实现*所示。

表 3-2. FlashPrg.c	函数接口的实现
-------------------	---------

```
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)
      Return Value:
                      0 - OK, 1 - Failed
*/
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);
                                                 /* Finished without Errors */
 return (0);
   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);
                                                 /* Finished without Errors */
 return (0);
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;
	}
}	
}	
retu	ırn adr+sz;

相关 SPI 驱动根据 GD32F4xx_Firmware_Library 和 GD25qxx.c 添加到 KEIL 工程中,所添加 的文件如 *图 3-2 移植 SPI 驱动和 GD25qxx 文件*所示。

图 3-2 移植 SPI 驱动和 GD25qxx 文件



3.3. 修改 FlashDevice 结构体

打开 FlashDev.c 文件,修改 FlashDevice 结构体中的相关内容,修改后代码如<u>表 3-3.</u> *FlashDevice 结构体实现*所示。

表 3-3. FlashDevice	结构体实现
--------------------	-------

truct FlashDevice const FlashDevice = {			
FLASH_DRV_VERS,	/* Driver Version, do not modify! */		
"GD25qxx",	/* Device Name */		
EXTSPI,	/* Device Type */		
0x0000000,	/* 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 Sectors */
 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. 编译生成 FLM 文件

打开魔术棒,进入 Output 页面,修改 Name of Executable 为 GD25Q16B,编译工程,生成 GD25Q16B.FLM 文件。如*图 3-3. 编译生成 GD25Q16B.FLM 文件*所示。



图 3-3. 编译生成 GD25Q16B.FLM 文件

3.5. 添加算法文件到 KEIL 工程中

将编译生成好的 GD25Q16B.FLM 拷贝到 KEIL 安装目录下, D:\Keil_527\ARM\PACK\GigaDevice\GD32F4xx_DFP\2.0.0\Flash,接着返回上级目录打开 GigaDevice.GD32F4xx_DFP.pdsc 文件,将其属性修改为可读写,找到 GD32F450IK,添加代



码如表 3-4. 修改 pdsc 文件代码 红色所示:

表 3-4. 修改 pdsc 文件代码

<!-- ************************* Device 'GD32F450IK' ******************************

<device Dname="GD32F450IK">

<memory id="IROM1" start="0x08000000" size="0x0300000" startup="1" default="1"/> <memory id="IRAM1" start="0x20000000" size="0x030000" init ="0" default="1"/> <memory id="IRAM2" start="0x10000000" size="0x010000" init ="0" default="0"/> <algorithm name="Flash/GD32F4xx_3MB.FLM" start="0x08000000" size="0x0300000" default="1"/> <algorithm name="Flash/GD25Q16B.FLM" start="0x00000000" size="0x01000000" default="1"/> </device>

在 KEIL 工程, 打开魔术棒 Utilities 页面 setting, 添加 GD25qxx 算法, 如<u>图 3-4. KEIL 中添加</u> GD25Qxx 下载算法</u>所示。



| Options for Target 'GD324 | 4501_EVAL' | | | | |
|--|--|---|--|---|---------------|
| ice Target Output List | ting User C/C++ | Asm Linker | Debug | Utilities | 1 |
| Configure Flash Menu Command | d | 05-4 | | | |
| Use Target Driver for Flash | Programming | 2 🖪 | Vse Debu | g Driver | |
| Use Debug Dr | river | Settings 1 | Update Ta | arget before Debu | gging |
| Init File: | | | | Edit | |
| C Use External Tool for Flash | Programming | | | | |
| Command: | | | | | |
| Arguments: | | | | | |
| Run Indeper | ndent | | | | |
| Configure Image File Processing | (FCARM): | | | | |
| Output File: | | Add Output File | to Group: | | |
| I | | Application | | | - |
| Image Files Root Folder: | | Г | Generate | Listing | |
| ex JLink/JTrace Target Driver | Setup | | | | |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
C Erase Full Chip
C Erase Sectors
C Do pot Erase | Setup
I
I
I
I
P
I
Program
I
Verify
I
I
I
I
I
I
I
I
I
I
I
I
I | RAM for Algor
Start: 0x200 | nithm
000000 | Size: 0x3000 | |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
Cond
C Erase Full Chip
C Erase Sectors
C Do not Erase
Download Function | Setup
P I Program
I Verify
I Reset and Run | RAM for Algor
Start: 0x200
tdd Flash Programming Algo | rithm
100000 | Size: 0x3000 | |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
C Erase Full Chip
© Erase Sectors
C Do not Erase
Programming Algorithm | Setup | RAM for Algoi
Start: 0x200 | rithm
000000
orithm
Bach Stre Dev | Size: 0x3000 | 5.52 |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
C Erase Full Chip
C Erase Sectors
C Do not Erase
Programming Algorithm
Description | Setup | RAM for Algoi
Start: 0x200
dd Fash Programming Algo
Coloma
Algo Starts Coloma
Algo Starts | rithm
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
200000
2000000
2000000
2000000
2000000
2000000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
200000000 | Size: 0x3000 | chage
2009 |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
C Erase Full Chi
© Erase Sectors
C Do not Erase
Programming Algorithm
Description | Setup
P IP Program
Verify
Reset and Run
Device Size Dev | RAM for Algori
Start: 0x200
Hot Flash Programming Algori
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Concent
Conce | rithm
2000000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
2000000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
200000
2000000
2000000
2000000
2000000
2000000
2000000
2000000
200000000 | Size: Dx3000 or. Too: Dees Size: Dees | chape
5.50 |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
C Erase Full Chi
C Erase Sectors
C Do not Erase
Programming Algorithm
Description | Setup
P IP Program
Verify
Reset and Run
Device Size Devi
3 | RAM for Algo
Start: 0x200
Add Flash Programming Algo
Concessor
Market Starts
Concessor
Market Starts
Concessor
Market Starts
Concessor
Market Starts
Concessor
Market Starts
Market Star | hithm
2000000
ordine
Set Set Deal
Set Set Deal
Set Set Set Set Set Set Set Set Set Set | Size: 0x3000 | 3.59 |
| ex JLink/JTrace Target Driver
bug Trace Flash Download
Download Function
C Erase Full Chi
C Erase Sectors
C Do not Erase
Programming Algorithm
Description | Setup
P IF Program
F Verify
Reset and Run
Device Size Devi
3
Add | RAM for Algo
Start: 0x200
dd flash Programming Algi
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Color
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Peccetor
Pe | rithm
200000
ordtwn
Teather
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State
State | Size: 0x3000 | |

图 3-4. KEIL 中添加 GD25Qxx 下载算法

3.6. 编译和下载

在 KEIL 中编译工程,生成.axf 文件,点击 Download 按钮实现文件的下载,如<u>图 3-5. KEIL 中</u> 编译和下载文件到 SPI Flash 所示,提示下载成功。



图 3-5. KEIL 中编译和下载文件到 SPI Flash



3.7. 测试和验证

为了确定文件是否成功下载到 GD25Q16BS Flash 中,参考 <u>Jflash-SPI 上位机配置与下载</u>, 通过 J-Flash SPI 上位机读取 Flash 中的数据,比较下载文件和读出文件是否相同,进行验证 测试。



4. 历史版本

表 4-1. 历史版本

| 版本号. | 描述 | 日期 |
|------|------|-------------|
| 1.0 | 首次发布 | 2021年04月30日 |



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