

通过 STM32CubeMX 制作外部 Flash 的烧 写驱动(.stdlr)

前言

目前,越来越多的应用需要扩展外部的 Flash 来满足存储需求。那么,在调试及批量生产的过程中,需要对外扩的 Flash 进行 烧录操作。由于 STM32 ST-LINK Utility 以及 STM32CubeProgrammer 中,对 Flash 支持的型号有限,只能覆盖一部分 MCU 和 Flash 的型号,无法完全满足客户的需求。而且,它提供的 external loader 的制作模板存在覆盖的芯片型号较少,且无法 前期 QSPI Flash 调试的问题。本文旨在提供一种通过 stm32CubeMX 制作 external Flash loader 的方法。客户可以根据自己 的型号,进行定制化的生成。本文中,以某客户实际使用的 MCU(STM32H750) 和 Flash(S25LP128F) 为例进行讲解。

准备工作

- 安装 STM32CubeIDE
- 安装 MCU 对应型号的 HAL 库

External loader 开发

External Loader 的开发分成三个部分,第一部分是使用 STM32CubeMX 进行工程的配置及生成。第二部分是外部 Flash 的驱 动调试,主要包含初始化,擦除,写入以及读出等操作。第三部分调用驱动函数进行 external loader 的生成,包括外部 Flash 信息的定义,包括 Flash 容量的大小,page 的大小,以及 Sector 相关的信息;第三步中 external loader 所需要的代码及对应 器件的驱动,可以在下面的 GitHub 仓库中获取,同时也欢迎大家将自己调试好的器件驱动提交到该仓库。 地址: https://github.com/WellinZHANG/External Loader

使用 STM32CubeMX 生成工程

此处,我们使用 STM32CubeIDE 集成的 STM32CubeMX 进行工程的配置和生成。

新建工程

选择对应的器件,填入工程名称,并选择存放位置:



101 Workspace - STM32CubelDE File Edit Source Refactor Navionte Search Bari			Designing ofal loa	der.pdfK
Information Center	IDE STM32 Project			
Contraction STM32CubeIDE Home	Target Selection Select STM32 target or STM32Cube example			
	MCII/MDII Selector Reard Selector Example S	alastar Crass Salastar		
Welcome	MCU/MPU Filters			
	් 📩 🔂 🔂 ථ	Features	Block Diagram Docs & Res	sources
	Part Number STM32H750VB ~	STM32H7 Series		
L⊕ L⊕	Carp		High-performance and DSP wi	ith DP-FPU Arm (
Start new Start new pro STM32 from an exist		STM32H750VB	of Flash memory, 1MB RAM, 4	80 MHz CPU, L1 o
project STM32Cube configuration	n Series 2		JPEG codec, HVV crypto, large	e set of periphera
(.ioc)	Line >	ACTIVE Active	01111101011010 (000). 3.003	
	Package >			
	Other >	MCUs/MPUs List: 1 item	Display similar items Marketing Stat×Unit Price for 10kU×	Board X Package
	Peripheral >	☆ STM32H750VB STM32H750V	BTx Active 3.685	LQFP100
Read STM32CubelD	E			
다. Getting Started with				
	_			
Explore What's New	/i			
	0			< B
	L			
IDE STM32 Project		— Г	1 X	
STIVISZ Project				
			INE	
Setup STM32 project			IDE	
D 1 1				
Project				
Project Name: H750_E	xternal_Loader			
Use default location				
Location: C:/Wor	rkspace	E	Browse	
Options				
Targeted Language				
Transferd Disease Trans				
Targeted Binary Typ	e			
Executable OS	tatic Library			
T . 15 . 17				
Targeted Project Ty	pe			
STM32Cube O	Empty			
(?) < Ba	ck Next >	Finish	ancel	

管脚配置:

按照硬件原理图选择好对应的 Flash 管脚,此处选择为 Bank2,配置如下图。注意调整管脚的速度为 High,同时使能 QSPI 的中断。FlashSize 值设置为 N, Flash 的大小配置 2^{N+1}。



								_
QUADSPI Mod	de and Configuration		QUAD	SPI Mode and	Configuration			
	Mode			Mode				
QuadSPI Mode Bank2 with Quad S Chip Select for Dual bank Disable	SPI Lines V	QuadS Chip S	SPI Mode Bank2 wit	h Quad SPI Lin Disable	es		~	
Con	figuration			Configurati	on			
Reset Configuration		Reset	Configuration					
📀 NVIC Settings 🛛 😔 GPI	O Settings 🛛 🥺 MDMA Settings	⊘ N\.	/IC Settings	🔗 GPIO Setti	nas		Settings	
✓ Parameter Settings	User Constants		Parameter Setti		· • • • • • • • • • • • • • • • • • • •	ser Const	ants	
Configure the below parameters :				.90	•••	on oonot	anto	
Q Search (CrtI+F) 🔇 🕥	0	Search S	Signals					
✓ General Parameters			0.00					
Clock Prescaler	4	Search (CrtI+F)					
Fifo Threshold	1	Pin N *	Signal on Pin C	ont GPIO ou	tGPIO mod	e GPIO I	ul Maxi	nu.
Sample Shifting	Sample Shifting Half Cycle	PB2	QUADSPn/a	n/a	Alternate.	. No pull	u Hiah	
Flash Size	27	PC11	QUADSPn/a	n/a	Alternate.	. No pull	u Hiah	
Chip Select High Time	3 Cycles	PE7	QUADSPn/a	n/a	Alternate.	. No pull	u Hiah	
Clock Mode	High	PE8	QUADSPn/a	n/a	Alternate.	. No pull	u High	
Flash ID	Flash ID 2	PE9	QUADSP n/a	n/a	Alternate.	. No pull	u High	
Dual Flash	Disabled	PE10	QUADSP n/a	n/a	Alternate.	. No pull	u High	



时钟配置

根据板子上的时钟源,进行对应的时钟配置,此处采用内部作为系统 PLL 的时钟源。





生成工程

切换到 Code Generator 选项卡,勾选 "Generate perioheral initialization as pair of '.c/.h'files per peripheral"选项。



Project	STM32Cube MCU packages and embedded software packs Copy all used libraries into the project folder Copy only the necessary library files Add necessary library files as reference in the toolchain project configuration file
	 Generated files ✓ Generate peripheral initialization as a pair of '.c/.h' files per peripheral □ Backup previously generated files when re-generating
	 Keep User Code when re-generating Delete previously generated files when not re-generated
Code Generator	HAL Settings Set all free pins as analog (to optimize the power consumption) Enable Full Assert
	Template Settings Select a template to generate customized code Settings

调试 QSPI 驱动

从开篇提到的 GitHub 仓库中获取相关的驱动代码。

git	2020/8/26 15:08	文件夹	
Loader_Files	2020/7/30 17:12	文件夹	
		2.01.1	
QSPI Drivers	2020/9/24 15:27	又件大	
QSPI testing	2020/7/30 17:12	文件夹	
README.md	2020/7/30 17:12	MD 文件	2 KB

第一步,将 External-Loaders\ QSPI testing 目录 mian_test.c 中的代码添加到工程中对应的 main.c 文件中。

> 本地磁盘 (C:) > Workspace > External-l	Loaders > QSPI testing		
へ 名称	修改日期	类型	大小
🗈 main test.c	2020/7/30 17:12	C Source	2 KB
📋 testbinary1M.bin	2020/7/30 17:12	BIN 文件	1,155 KB



/* USER CODE BEGIN 0 */
#include <string.h>
#define SECTORS_COUNT 100
/* USER CODE END 0 */

```
/* USER CODE BEGIN 2 */
  uint8_t buffer_test[MEMORY_SECTOR_SIZE];
uint32_t var = 0;
  CSP_QUADSPI_Init();
  for (var = 0; var < MEMORY_SECTOR_SIZE; var++) {</pre>
         buffer_test[var] = (var & 0xff);
  for (var = 0; var < SECTORS_COUNT; var++) {</pre>
         if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,
                      (var + 1) * MEMORY_SECTOR_SIZE - 1) != HAL_OK) {
               while (1)
                      ; //breakpoint - error detected
         if (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE,
                      sizeof(buffer_test)) != HAL_OK) {
               while (1)
                      ; //breakpoint - error detected
  if (CSP_QSPI_EnableMemoryMappedMode() != HAL_OK) {
         while (1)
               ; //breakpoint - error detected
  for (var = 0; var < SECTORS COUNT; var++) {</pre>
         if (memcmp(buffer_test,
                      (uint8_t*) (0x9000000 + var * MEMORY_SECTOR_SIZE),
                      MEMORY_SECTOR_SIZE) != HAL_OK) {
               while (1)
                      ; //breakpoint - error detected - otherwise QSPI works properly
```



/* USER CODE END 2 */

注意添加代码时保留 USER CODE BEGIN x 和 USER CODE END x 标签,否则添加的代码会在 Cube MX 重新生成代码时被 覆盖掉。

第二步,将\External-Loaders\QSPI Drivers\IS25LP128F 目录下的 QSPI 的 HAL 驱动文件替换工程中对应的文件。

> 本地磁盘 (C:) > Workspace > Exte	ernal-Loaders > QSPI Drive	ers > IS25LP128F		~
	修改日期	类型	大小	
🖻 quadspi.c	2020/9/15 14:25	C Source	16 KB	
🖻 quadspi.h	2020/9/15 14:26	C/C++ Header	2 KB	

第三步,打开 CubeMX,重新生成工程。因为为了适配所有系列的 QSPI 接口,GitHub 所提供 QSPI 的 HAL 驱动中,没有提供相应的 QSPI 外设配置函数,需要通过 CubeMX 来生成。

/*IS25LP128F memory parameters*/		
#define MEMORY_FLASH_SIZE	0x8000000	/* 128 MBits => 16MBytes */
#define MEMORY_BLOCK_SIZE	0x10000	/* 1024 sectors of 64KBytes */
#define MEMORY_SECTOR_SIZE	0x1000	/* 16384 subsectors of 4kBytes */
#define MEMORY_PAGE_SIZE	0x100	/* 262144 pages of 256 bytes */
/*IS25LP128F commands */		
#define WRITE_ENABLE_CMD 0x06		
#define READ_STATUS_REG_CMD 0x05		
#define READ_FLAG_STATUS_REG_CMD 0x05/	//0x70	
#define WRITE_STATUS_REG_CMD 0x01		
#define SECTOR_ERASE_CMD 0x20		
#define CHIP_ERASE_CMD 0xC7		
#define QUAD_IN_FAST_PROG_CMD 0x38		
#define READ_CONFIGURATION_REG_CMD 0x6	61//0x15	
#define WRITE_CONFIGURATION_REG_CMD 0x	:65//	
#define QUAD_READ_IO_CMD 0xEC		
#define QUAD_OUT_FAST_READ_CMD 0x6B		
#define QPI_ENABLE_CMD 0x35		
#define <u>DUMMY_CLOCK_CYCLES_READ_QUAD</u> 8	3	
#define RESET_ENABLE_CMD 0x66		
#define RESET_EXECUTE_CMD 0x99		
#define DISABLE_QIP_MODE 0xf5		
/* USER CODE END Prototypes */		

第四步,如上图所示由于每个型号的 FLASH 的控制指令略有差别,所以在此步需要根据自己选用的 QPSI FLASH 器件进行 调整。同时需要进行调试 QSPI 的读写是否正常。如果 QSPIF lash 读写不正常,那么在下图中标注的位置添加断点,就可以 排查是哪一个环节出错,进一步对驱动进行调整。



<pre>/* USER CODE BEGIN 2 */ uint8_t buffer_test[MEMORY_SECTOR_SIZE]; uint32_t var = 0; CSP_QUADSPI_Init(); for (var = 0; var < MEMORY_SECTOR_SIZE; var++) { buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE, (var + 1) * MEMORY_SECTOR_SIZE - 1) != HAL_OK) { while (1) ; //breakpoint - error detected j if (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE, sizeof(buffer_test)) != HAL_OK) { while (1) ; //breakpoint - error detected ; while (1) ; //breakpoint - error detected ; //breakpoint - error detecte</pre>
<pre>uint8_t buffer_test[MEMORY_SECTOR_SIZE]; uint8_t buffer_test[MEMORY_SECTOR_SIZE]; uint32_t var = 0; GSP_QUADSPI_Init(); for (var = 0; var < MEMORY_SECTOR_SIZE; var++) { buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>unitS_t conter_test[netWork_Sterox_Ster]; uintS_t var = 0; CSP_QUADSPI_Init(); for (var = 0; var < MEMORY_SECTOR_SIZE; var++) { buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>Government of the set of the</pre>
<pre>GSP_QUADSPI_Init(); for (var = 0; var < MEMORY_SECTOR_SIZE; var++) { buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>S CSP_QUADSPI_Init(); for (var = 0; var < MEMORY_SECTOR_SIZE; var++) { buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>6 7 7 7 7 7 7 7 7 7 7 7 7 8 7 7 8 9 9 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</pre>
<pre>for (var = 0; var < MEMORY_SECTOR_SIZE; var++) { buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>buffer_test[var] = (var & 0xff); } for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>9 } 9 for (var = 0; var < SECTORS_COUNT; var++) { 2 if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE, 4</pre>
<pre>for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>for (var = 0; var < SECTORS_COUNT; var++) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>if (val = 0; val < 5_could, val +) { if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>if (CSP_QSPI_EraseSector(var * MEMORY_SECTOR_SIZE,</pre>
<pre>if (CSP_QSPI_EraseSector(var' MEMORY_SECTOR_SIZE,</pre>
<pre>4</pre>
<pre>while (1) ; //breakpoint - error detected if (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE, sizeof(buffer_test)) != HAL_OK) { while (1) ; //breakpoint - error detected } } if (CSD_CSD_Ecologient ended() is (WE CO) f </pre>
<pre>while (1) ; //breakpoint - error detected ; //breakpoint - error detected ; //breakpoint - error detected while (1) ; //breakpoint - error detected ; //breakpoint - error detected ; } if (CFD error for the due to be any or) for </pre>
<pre>; //breakpoint - error detected if (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE,</pre>
<pre> } if (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE, sizeof(buffer_test)) != HAL_OK) { while (1)</pre>
<pre>9 9 1 (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE, 1 sizeof(buffer_test)) != HAL_OK) { 9 1 while (1) 1 ; //breakpoint - error detected 1 1 2 3 } 3 4 ; //breakpoint - error detected 3 5 } 6 7 } 8 1 ((CSD_SCD_SCD_SCD_SCD_SCD_SCD_SCD_SCD_SCD_</pre>
<pre>0 if (CSP_QSPI_WriteMemory(buffer_test, var * MEMORY_SECTOR_SIZE, 1 sizeof(buffer_test)) != HAL_OK) { 2 while (1) 4 ; //breakpoint - error detected 5 } 6 7 } 8 8 16 (CSP_QSPI_Enclutionentianentiteentianentianentianentianentianentianentianentianentianent</pre>
<pre>sizeof(buffer_test)) != HAL_OK) { while (1) ; //breakpoint - error detected } f f f f f f f f f f f f f f f f f f</pre>
while (1) ; //breakpoint - error detected ; //breakpoint - error detected ; //breakpoint - error detected
<pre>while (1)</pre>
<pre>4 ; //breakpoint - error detected 5 } 6 7 } 8 8</pre>
<pre></pre>
5 } 6 7 } 8 0 16 (552 552 55 b) the set the de () d = 100 50 f
o 7 } 8
$a = \frac{1}{2} \left(ccp - ccp - c-b \right) - M_{abs} - dM_{abs} - dM_{abs$
<pre>9 1f (LSP_QSP1_EnableMemoryMappedMode() != HAL_UK) {</pre>
<u> 2</u>
1 while (1)
2 ; //breakpoint - error detected
A
$\int_{-\infty}^{\infty} f(y_{2}) = \theta(y_{2}) + \int_{-\infty}^{\infty} f(y_{2}) + \int_{-\infty}^{\infty$
6 IT (memcmp(butter_test,
/ (uint8_t*) (0x9000000 + Var * MEMORY_SECTOR_SIZE),
MEMORY_SECTOR_SIZE) 1= HAL_OK) {
9 while (1)
//hreaknoint - error detected - otherwise OSPT works properly
, //breakpoint - citor detected - otherwise Osti works property
, //b/eakpoint - error detected - otherwise gari works property
<pre>></pre>

修改配置生成 QSPI Loader

完成 QSPI 的驱动调试之后,我们需要添加生成 external Loader 所需要的代码并修改对应的配置。

第一步,添加对应的代码,存放在\External-Loaders\Loader_Files 目录下,由于 H7 和其他的系列的 linker file 有所区别,所 以此处分为 H7 和 others 两个文件夹进行存放。我们此处选用 H7 目录下的问题件。将所有的文件添加到工程中。

> 本地磁盘 (C:) > Workspace > External-	Loaders > Loader_Files	;
へ 、	修改日期	業型 大小
H7 device	2020/7/30 17:12	文件夹
📊 other devices	2020/7/30 17:12	文件夹



→ 本地磁盘 (C:) → Workspace → External-L	.oaders > Loader_Files	> H7 device		~	Ū
~ 名称	修改日期	类型	大小		
🗈 Dev_Inf.c	2020/7/30 17:12	C Source	1 KB		
🖻 Dev_Inf.h	2020/7/30 17:12	C/C++ Header	1 KB		
linker.ld	2020/7/30 17:12	LD 文件	4 KB		
ि Loader Src.c	2020/7/30 17:12	C Source	7 KB		

第二步,修改 Dev_Inf.c 中的 name 为你想设置的名称,一般设置为 MCU+Flash 名称

8	/* This structure contains information used by	/ ST-LINK Utility to program and erase the device */				
9	<pre>#if defined (ICCARM)</pre>	#if defined (ICCARM)				
10	<pre>root struct StorageInfo const StorageInfo =</pre>	= {				
11	#else					
12	<pre>struct StorageInfo const StorageInfo = { mondif</pre>					
14	"STM32H750VB+IS25LP128F",	// Device Name + version number				
15	NOR_FLASH,	// Device Type				
16	0x90000000,	// Device Start Address				
17	MEMORY_FLASH_SIZE,	// Device Size in Bytes				
18	MEMORY_PAGE_SIZE,	// Programming Page Size				
19	0xFF,	<pre>// Initial Content of Erased Memory</pre>				
20						
21	// Specify Size and Address of Sectors (view example below)					
22	{ { (MEMORY FLASH SIZE / MEMORY SECTOR SIZE), // Sector Numbers,					
23	(uint32 t) MEMORY SECTOR SIZE }, //Sector Size					
24	· · · · · · · · · · · · · · · · · · ·					
25	{ 0x00000000, 0x00000000 } } }					

第三步,修改 Linker 的配置,通过 "project" -> "Properties" 打开设置页。将红框内的 ld 文件修改为 linker.ld.(已经在第一步中添加到工程)。





第四步,修改配置,编译后生成 stdlr 文件。通过 "project" -> "Properties" 打开设置页,在 "post build steps" 处添加如 下指令:

md.exe /C copy/Y	\${BuildArtifactFileBaseName}.elf" "\SIM32H/50_1S25LP128F.stldr"
Properties for Haier_Exte	rnal_Loader — 🗆 X
	Settings $\Diamond \checkmark \circlearrowright \checkmark \checkmark$
> Resource	
Builders	
✓ C/C++ Build	Configuration: Debug [Active]
Build Variables	
Environment	🖉 Build Steps 🕕 Build Addingt 🖹 Dinga Demons 🙆 Energy Demons
Logging	Build Artifact M Binary Parsers V Error Parsers
Settings	Pre-build steps
> C/C++ General	Command:
CIVISIS-SVD 设直 Droject References	×
Run/Debug Settings	Description:
Run Debug Settings	
	×
	-Post-build steps
	Command:
	cmd.exe /C copy/Y "\${BuildArtifactFileBaseName}.elf" "\STM32H750 IS25LP128F.stldr"
	Description:
	×



最后,编译便可在工程目录下生成对应的 stdlr 文件。将其复制到 STM32CubeProgrammer 安装目录下的 extral loader 文件 夹下,便可使用。



总结

通过该方法可以快速的生成一个外部烧写脚本对外部的 QSPI FLASH 进行烧录。

重要通知 - 请仔细阅读

意法半导体公司及其子公司("ST")保留随时对ST产品和/或本文档进行变更、更正、增强、修改和改进的权利,恕不另行通知。买方在订货之前应获取关于ST产品的最新信息。ST产品的销售依照订单确认时的相关ST销售条款。

买方自行负责对ST 产品的选择和使用, ST 概不承担与应用协助或买方产品设计相关的任何责任。

ST 不对任何知识产权进行任何明示或默示的授权或许可。

转售的ST 产品如有不同于此处提供的信息的规定,将导致ST 针对该产品授予的任何保证失效。

ST 和ST 徽标是ST 的商标。所有其他产品或服务名称均为其各自所有者的财产。

本文档中的信息取代本文档所有早期版本中提供的信息。

© 2015 STMicroelectronics - 保留所有权利