Windows 下 STM32 单片机的 eclipse 编译环境搭建

英创公司开发的 ETA321 单片机模块,是基于 STM32F103RC 单片机设计的实时任务 处理单元模块。

ETA321 模块作为客户端,使用 USB 与英创公司的 ARM 工控主板进行数据传输、通讯,根据工控主板应用程序发出来的指令,执行相应的实时任务处理,如实现:电机控制、数据采集、高速 D/A 输出、状态采集保护等。另外,用户可以基于英创公司构建好的 USB 通讯结构,利用 KEIL、IAR、eclipse 等集成编译环境,在 ETA321 模块上编写自己特殊的应用程序。英创公司推荐使用 eclipse 编译环境。

由于 eclipse 软件编译工具是开源、免费的工具,在 windows 环境下编译 linux 应用软件的用户也比较多,因此在 windows 环境中,搭建 eclipse 的 STM32 编译环境,就变得非常有意义:可以很好地与 linux 编译环境进行整合;不用担心开发软件的版权问题; eclipse 编译环境也能支持各种调试工具。

因此,使用 eclipse 编译环境完全替换 KEIL、IAR 等需要授权才能使用工具,是非常好的选择。但是搭建基于 eclipse 的 STM32 的开发环境,需要好几个软件与插件,且相应的版本需要能兼容,才能正确搭建好开发环境。

为了协助客户快速搭建 eclipse 编译环境,英创公司编写了这篇文章,详细说明了整个 eclipse 环境搭建、工程建立、目标代码烧写、以及使用 JLink 进行软件调试的方法。

所需要的软件或插件如下:

- JAVA
- eclipse C/C++
- gcc-arm-none-eabi, 交叉编译工具链
- CDT,对 cortex-M的支持的编译工具以及对 J-Link 调试工具的支持
- J-LINK 驱动

开发/烧写 STM32 所需要的软件/工具:

● STM32 Flashloader 串口烧写工具

为了方便客户更加简单、方便地搭建 ETA321 的开发环境,英创公司下载了所需的工具 软件: JAVA、eclipse C/C++、gcc-arm-none-eabi、STM32-Flashloader 等,可以直接利用。

文件(F) 编辑(E) 查看(V) 书签(A) 工具(T) 帮助	(H)				
多 E: と <	编译环境搭建\STN	132 eclipse W	in7环境搭建软	件.zip\	~
名称	大小	压缩后大小	修改时间	创建时间	访问即
L eclipse	670 019 7	331 015 4	2017-10-1	2017-09-2	2017
E (J-LINK Driver)JLink_Windows_V620c.exe	26 735 552	26 712 416	2017-09-2	2017-09-2	2017
🚺 ST-flashloader.zip	34 997 919	34 997 919	2017-09-2		
I (JAVA) jdk-8u144-windows-i586.exe	200 214 0	198 573 8	2017-09-2	2017-09-2	2017
<					>

图 1、eclipse 编译环境软件工具包

下面则详细地描述 eclipse 编译环境塔建方法。

- 一、安装 eclipse 编译环境
- 1、安装 JAVA

打开工具包中的(JAVA)jdk-8u144-windows-i586.exe 进行默认安装即可。

	52
於 Java SE Development Kit 8 Update 144 - 安装程序	_ 43_
Java Java	
欢迎使用 Java SE 开发工具包 8 Update 144 的安装向导	
本向导将指导您完成 Java SE 开发工具包 8 Update 144 的安装过程。	
Java Mission Control 分析和诊断工具套件现在作为 JDK 的一部分提供。	
下一步(N) >	取消

图 2、启动并安装 JAVA

Java 安装 - 进度	
Java Bava	
状态: 安装 Java	
ATMs, Smartcards, POS Terminals, Blue Set Top 3 MBIII Routers, 3 BIII Automotion Devices Run	u-ray Players, PCs Servers, Switches Devices Lottery Java controls
Java #1 Development Platform	ORACLE

图 3、JAVA 安装中...

100	
0	🕼 Java SE Development Kit 8 Update 144 - 完成
	E Java
	Java SE Development Kit 8 Update 144 已成功安装
	单击"后殡步骤"访问教程, API 又相, 开发入负指南, 发布说明及更多内谷, 帮助您开 始使用 JDK。
	后续步骤(N)
	大(ग)(C)

图 4、JAVA 安装完成

现阶段,不建议安装 JAVA 9.0 版本。因为要实现 JAVA 9.0 对 eclipse 的支持,需要额外的补丁包,所以相对麻烦一些。

2、安装 eclipse

复制 eclipse 文件夹到开发用的计算机中(任意位置均可),如: C:\program file(x86) 目录(如果系统是 32 bit,则 C:\program file)。

	ogram Files (x86) 🕨
组织 🔹 🧱 打开 包含到库中 🔹	共享▼新建文件夹
 ☆ 收藏夹 ▶ 下载 ■ 桌面 ◎ 最近访问的位置 ▶ 照片流 	▲ 名称 eclipse Common Files Java Internet Explorer

图 5、将 eclipse 复制到 Program Files(x86)目录下

3、设置交叉编译工具链的环境变量

从 eclipse 目录下,找到交叉编译工具链 gcc-arm-none-eabi 的路径,并复制下来。



打开"系统属性"->"高级"配置页面,再点击"环境变量",进入环境变量设置页面。

受里	值	
HOME	C:\SPB_Data	
DA_PLUGIN_PATH	%CDSROOT%\Share\oaPlugIns	E
ATH DOOTDID	%CDSROOT%\tools\libutil\bin;%CD	
1919_VOOLDIK	c. (artera (14. o (quar tus (sope_bur	*
统变量(S)		
统变量(S)	<i>t</i> ±	
统变量(S) 变量	值 c:\Program Files (x86)\#MD #PP\	
统变量(S) 变量 AMDAPPSDKROOT ComSnec	值 c:\Program Files (x86)\AMD APP\ C:\Windows\system32\cmd.exe	
统变量(S) 变量 AMDAPPSDKROOT ComSpec FP_NO_HOST_C	值 c:\Program Files (x86)\AMD APP\ C:\Windows\system32\cmd.exe NO	
统变量(S) 变量 AMDAPPSDKROOT ComSpec PP_NO_HOST_C NUMBER_OF_PR	值 c:\Program Files (x86)\AMD APP\ C:\Windows\system32\cmd.exe NO 4	•

图 7、环境变量设置页面

在用户环境变量中找到 PATH 项,点击编辑,将工具链的路径添加到 PATH 参数中,点击确认退出。

编辑用户变量	
变量名(N):	РАТН
变量值(Ⅴ):	(x86)\eclipse\gcc-arm-none-ea
	确定 耶
变量	
AMDAPPSDKROOT	c:\Program Files (x86)\AMD APP\
FP_NO_HOST_C	C:\Windows\system32\cmd.exe NO
NUMBER_OF_PR	4
	「新建(₩) 编辑(I) 删

图 8、在用户环境变量的 PATH 变量名中,添加工具链的路径

环境变量添加完成后,可以从系统的 CMD 命令提示符窗口中,输入 path 指令,检查 环境变量是否添加成功。



- 图 9、检查工具连的环境变量
- 4、安装 CDT(C/C++ Development Tooling)

进入 eclipse 目录,双击 eclipse.exe 启动 eclipse 软件,这时设置 eclipse 的工作区路 径,可以系统中的任意路径均可。设置好以后,点击"launch"。

🗢 Eclipse Lau	incher	
Select a dire	ectory as workspace the workspace directory to store its pr	eferences and development artifacts.
<u>W</u> orkspace:	D:\Project\software\STM32_XXX	<u>B</u> rowse
✓ <u>U</u> se this a	s the default and do not ask again	Launch Cancel

图 10、设置 eclipse 工作区

软件启动完成后,在 eclipse 的"help"菜单中,选择"install New software"

Window He	elp		-
6	Welcome		
ne Ecli [®]	Help Contents Search Show Contextual Help		
đ	Show Active Keybindings Tips and Tricks Report Bug or Enhancement Cheat Sheets	Ctrl+Shift+L	$\langle \rangle$
ting pro	Eclipse User Storage Perform Setup Tasks	*	7
24	Check for Updates		
9	Install New Software		
n settir	Eclipse Marketplace		
contested	About Eclipse		

图 11、选中 help->Install New Software

然后在"Work With:"中输入链接"http://gnu-mcu-eclipse.netlify.com/v4-neon-updates" 并回车,将会列出该 CDT 所包含的所有工具列表。在列出来的选项中,全部选择,点击"next"。

vailable Software	
Check the items that you wish to install.	Ó
ork with: http://gnu-mcu-eclipse.netlify.com/v4-neon-updates	Add Manage
pe filter text	
lame	Version
GNU ARM & RISC-V C/C++ Cross Development Tools	
GNU MCU C/C++ ADuCM360 Project Template	1.1.5.201707111115
🔽 🖗 GNU MCU C/C++ ARM Cross Compiler	2.5.2.201707111115
🔽 🖗 GNU MCU C/C++ CodeRed Debug Perspective	1.1.2.201707111115
GNU MCU C/C++ Documentation (Placeholder)	1.1.2.201707111115
GNU MCU C/C++ Freescale Project Templates	2.2.9.201707111115
🔽 🖗 GNU MCU C/C++ Generic Cortex-M Project Template	1.4.3.201707111115
🔽 🚱 GNU MCU C/C++ J-Link Debugging	4.1.5.201707111115
🔽 🖗 GNU MCU C/C++ OpenOCD Debugging	4.1.5.201707111115
🔽 🖗 GNU MCU C/C++ Packs (Experimental)	2.2.2.201707111115
🔽 🖗 GNU MCU C/C++ PyOCD Debugging	1.1.4.201707111115
🔽 🖗 GNU MCU C/C++ QEMU Debugging	3.1.5.201707111115
GNULMCULC/C++ RISC-V Cross Compiler	2 5 2 201707111115
Show only the latest versions of available software <u>G</u> roup items by category What is a Show only software applicable to target environment	ems that are already installed already installed?
<u>Contact all update sites during install to find required software</u>	
? < <u>B</u> ack	Next > Einish Cancel
3 < Back 图 12、选中所有列	Next > Einish Cancel 表

🖨 Install	
Review Licenses Licenses must be reviewed and accepted before the sof	tware can be installed.
Licenses:	License text:
Eclipse Foundation Software User Agreement Eclipse Foundation Software User Agreement	Eclipse Foundation Software User Agreement April 9, 2014 Usage Of Content THE ECLIPSE FOUNDATION MAKES AVAILABLE SOFTWARE, DOCUMENTATION, INFORMATION AND/OR OTHER MATERIALS FOR OPEN SOURCE PROJECTS (COLLECTIVELY "CONTENT"). USE OF THE CONTENT IS GOVERNED BY THE TERMS AND CONDITIONS OF THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. BY USING THE CONTENT, YOU AGREE THAT YOUR USE OF THE CONTENT IS GOVERNED BY THIS AGREEMENT AND/OR THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS OF THIS AGREEMENT AND THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW. IF YOU DO NOT AGREE TO THE TERMS AND CONDITIONS OF THIS AGREEMENT AND THE TERMS AND CONDITIONS OF ANY APPLICABLE LICENSE AGREEMENTS OR NOTICES INDICATED OR REFERENCED BELOW, THEN YOU MAY NOT USE THE CONTENT. Applicable Licenses Unless otherwise indicated, all Content made available by the I accept the terms of the license agreements I do not accept die terms of the license agreements
0	< Back Next > Finish Cancel

图 13、接受 license 后开始安装

在 eclipse 的右下角可以看到安装进度,双击小的进度条,则可以打开安装信息。在安装过程中,如果有安全警告,点击"Install anyway"继续完成安装。



图 14、安装过程中的安全警告

当 CDT 安装完后,会提示软件重启,这时点击"Restart Now",重新启动 eclipse。



图 15、CDT 安装完成,需要重新启动软件

到此,在 Windows7 系统下,STM32 的 eclipse 编译环境已经搭建完成,接下来就可以建立第一个应用程序工程了。

二、建立 eclipse 应用工程程序

eclipse 编译环境安装完成以后,就可以开始编写应用程序了,我们以 ETA321 模块上的 LED 灯 D1 闪烁为例子,举例说明建立简单的应用工程。

1、启动 eclipse,从 file->new 中选择 C/C++ Project。

0 :	STM32_XXX - Eclipse				
File	Edit Source Refactor Navigate	Search Project	Ru	ın Window Help	
(New	Alt+Shift+N ▸	C++	Makefile Project with Existing Code	
	Open File Open Projects from File System		C++	C++ Project C Project	
	Close Close All	Ctrl+W Ctrl+Shift+W		C/C++ Project Project	
	Save Save As Save All Revert	Ctrl+S Ctrl+Shift+S		Convert to a C/C++ Autotoois Project Convert to a C/C++ Project (Adds C/C++ Nature) Source Folder Folder Source File	
2	Move Rename Refresh Convert Line Delimiters To	F2 F5		Header File File from Template Class Task	
0	Print	Ctrl+P	Ľ	Example	
èn.	Import			Other	Ctrl+N

图 16、建立工程

2、选择 C++ Managed Build, 点击"next"。



3、输入工程名,以及选择 Project type 为"STM32F10x C/C++ project", 右侧 Toolchains 选择为"ARM Cross GCC",点击"next"。

Use default location	
Location: D:\Project\software\STM32_XXX\ET	FA321_LED Browse
Choose file system: default 💌	
Project type:	Toolchains:
GNU Autotools	ARM Cross GCC
Executable	
Empty Project	
Hello World C++ Project	
Hello World ARM C++ Project	
Hello World RISC-V C++ Project	
ADuCM36x C/C++ Project	
Hello World ARM Cortex-M C/C++ Project	t
Freescale Kinetis KLxx C/C++ Project	
STM32F0xx C/C++ Project	
STM32F10x C/C++ Project	
• STM32F2xx C/C++ Project	
• STM32F3xx C/C++ Project	
• STM32F4xx C/C++ Project	
STM32F7xx C/C++ Project	
Shared Library	
Static Library	
Makenie project	

图 18、工程编译参数选择

4、设置芯片属性, ETA321 采用的是高密度器件, Flash 大小为 256KB, RAM 大小为 48KB, 根据该参数进行设置,点击"next"。

Chip family:	STM32f10x High Density
Flash size (kB):	256
RAM size (kB):	48
External clock (Hz):	8000000
Content:	Blinky (blink a led)
Use system calls:	Freestanding (no POSIX system calls)
Trace output:	Semihosting DEBUG channel
Check some warnings	
Check most warnings	
Enable -Werror	
Use -Og on debug	
Use newlib nano	
Exclude unused	
Use link optimizations	

5、设置工程所用的文件夹,可以默认即可,点击"next"。

Define the project fol	dars and other options	
Define the projection	ders and other options.	-
Include folder:	include	
Source folder:	src	
System folder:	system	
CMSIS library folder:	cmsis	
C library folder:	newlib	
Linker scripts folder:	ldscripts	
٢		
	< Back Next > Finish	Cancel

6、选择工程中需要编译的项目,一般情况下,"Debug"与"Release"都会默认选择,点击"next"。

Select platforms	and configurations you wish to deploy on	
Project type: Toolchains: Configurations:	Executable ARM Cross GCC	
 ✓ Some Debug ✓ Some Release 		Select all Deselect all
Use "Advanced Additional confi Use "Manage co	settings" button to edit project's properties. gurations can be added after project creation. nfigurations" buttons either on toolbar or on prop	Advanced settings.
?	< Back Next >	Finish Cancel

7、设置工具链。如果设置了工具链环境变量且系统重启生效,则"Toolchain path"会自动填写完成。如果这里为空,可以点击"Browse"按钮,手动选择工具链的目录即可,即上面所说的工具链路径,最后点击"finish"完成工程建立。

Select the toold	s Toolchain	Ď
Select the toolc		
Toolchain name:	GNU Tools for ARM Embedded Processors (arm-none-eabi-gcc)	-
Toolchain path:	C:\Program Files (x86)\eclipse\gcc-arm-none-eabi\bin	Browse
?	< Back Next > Finish	Cancel
3	图 22、配置工具链路径	

8、工程建立完成后,需要进行编译工具的配置。我们提供的 eclipse 工具包,有两个 编译工具可以选择: CDT Internal Builder 或 Gnu Make Builder,使用其中一个即可。配置 编译工具的方法如下:

首先,在 eclipse 环境中,选中所建立的工程,点击菜单中的"project"-> "Properties", 打开配置页面。

d mai	n ci	an	-	Open Project	
1.9	/*	P		Close Project	
2	*	Th			Ctol. D
3	*		010	Bulla All	Ctri+B
4	*	Co		Build Configurations	+
5	*			Build Project	
6	*	Pe		Build Working Set	
7	*	ob		Clean	
8	*	fi		Build Automatically	
9	*	re		j	
10	*	со		Build Targets	+
11	*	se		C/C Index	
12	*	th		C/C++ Index	
13	*	co	<	Properties	
1.4	*	1	-	the second se	

图 23、进入工程属性配置

a)、选择 CDT Internal Builder 编译工具

在弹出的配置页面中,展开左侧的"C/C++ Build"选项,点击"Tool Chain Editor",在右边的参数"Current Builder"下拉列表中,选择 CDT Internal Builder,然后点击"Apply and Close" 完成设置。

type filter text	Tool Chain Editor	
 > Resource Builders • C/C++ Build Build Variables Environment Logging Settings Tool Chain Editor > C/C++ General Linux Tools Path > MCU Project References Run/Debug Settings > Task Repository Task Tags > Validation WikiText 	Tool Chain Editor Configuration: Debug [Active] Image: Display compatible toolchains only Current toolchain: ARM Cross GCC Current builder: Gnu Make Builder Autotools Makefile Generator Used tools Gnu Make Builder GNU ARM Cross Assembler GNU ARM Cross C++ Compiler GNU ARM Cross C++ Linker GNU ARM Cross C++ Linker GNU ARM Cross Create Flash Image GNU ARM Cross Create Listing GNU ARM Cross Print Size	Manage Configurations Manage Configurations Select Tools
		Restore <u>D</u> efaults <u>Apply</u>

图 24、选择编译工具为"CDT Internal Builder"

b)、选择 Gnu Make Builder 编译工具

从上图中可以看出, eclipse 也可以使用 Gnu Make Builder 编译工具。在我们的 eclipse

工具包中,已经包含了该编译工具,在 eclipse 文件夹中的"GNU MCU Eclipse"就是该工具

包,因此可以配置使用该编译工具。使用该工具的配置方法如下。

首先确认编译工具的路径并复制下来。

	- Restantions			. D X
C:\Program Files (x86)\e	clipse\GNU MCU Eclipse\Build Tools\2.9-	20170629-1013\bin	搜索 bin	Q
组织▼ 包含到库中▼ 共享▼	新建文件夹		•	
🚖 收藏夹	▲ 名称	修改日期	类型	大小
🔰 下载	busybox.exe	2017/6/29 18:27	应用程序	418
重 桌面	🔳 echo.exe	2017/6/29 18:27	应用程序	418
🐉 最近访问的位置	💽 make.exe	2017/6/29 18:27	应用程序	202
1. 照片流	🔳 mkdir.exe	2017/6/29 18:27	应用程序	418
	III rm.exe	2017/6/29 18:27	应用程序	418
篇库	sh.exe	2017/6/29 18:27	应用程序	418

图 25、GNU Make Builder 路径确认 <

点击菜单中的"project"-> "Properties",选中左侧的"C/C++ Build"下"Environment",会 在右侧列出相应的环境变量参数。单击选择右侧的"PATH"项,再点击"Edit"按钮,进行参数 配置。

pe filter text	Environment		
Resource Builders C/C++ Build Build Variables Environment Logging	Configuration: Environment v	Debug [Active]	Manage Configurat
Logging Settings Tool Chain Editor C/C++ General Linux Tools Path MCU Project References Run/Debug Setting: Task Repository Task Tags Validation WikiText	Variable CWD PATH PWD	Value D:\Project\software\STM32_XXX\ETA321_LED\De C:\Program Files (x86)\eclipse\gcc-arm-none-eab D:\Project\software\STM32_XXX\ETA321_LED\De "" iables to native environment ive environment with specified one	Origin BUILD SYSTEM BUILD SYSTEM BUILD SYSTEM Un
• III			Restore Defaults A

图 26、选择"Environment" 配置项

在弹出的"Edit variable"参数修改页面中,添加 GNU Make Builder 工具的路径到 PATH

环境变量中。

X
PATH
\Build Tools\2.9-20170629-1013\bin Variables

图 27、在 PATH 参数中,添加 GNU Make Builder 工具路径

最后,在左侧的"C/C++ Build"选项,点击"Tool Chain Editor",在右边的参数"Current Builder"下拉列表中,选择"Gnu Make Builder",然后点击"Apply and Close"完成设置。

type filter text	Tool Chain Editor		↓ ↓ ▼
 Resource Builders C/C++ Build Build Variables Environment 	Configuration: Debu	ug [Active]	▼ Manage Configurations
Logging Settings Tool Chain Edito	☑ Display compatibl Current toolchain:	le toolchains only RM Cross GCC	
 > C/C++ General Linux Tools Path > MCU > Project References Run/Debug Setting: > Task Repository Task Tags > Validation WikiText 	Current builder: Used tools GNU ARM Cross A GNU ARM Cross C GNU ARM Cross C GNU ARM Cross C GNU ARM Cross C GNU ARM Cross C	inu Make Builder utotools Makefile Generator DT Internal Builder nu Make Builder Ssembler Compiler + + Compiler Linker ++ Linker rchiver	 Select Tools
	GNU ARM Cross Ci GNU ARM Cross Ci GNU ARM Cross Pr	reate Flash Image reate Listing int Size	-
<			Restore Defaults Apply Apply and Close Cancel

图 28、选择编译工具为"Gnu Make Builder"

10、由于 ETA321 模块上的 LED 灯使用的是 PB2 引脚,与建立的工程中默认使用的 GPIO 不同,所以需要修改代码,与 ETA321 的 GPIO 对应。需要修改的定义文件为 BlinkLed.h 文件,其中的 BLINK_JPORT_NUMBER 参数更改为 1,BLINK_PIN_NUMBER 参数更改 为 2。

ြဲ Project Explorer 🛛 🚆 🗖	i main.cpp	
🖻 😫 👘 🗢	33 // LED definitions	
▷ 💕 ETA321_LED	34 35 // Adjust these definitions for your 36 37=// <u>Olimex</u> STM32-H103 definitions (thu 38 // (SEGGER J-Link device name: STM321 39 40 // <u>Devt numbers:</u> 0=A, 1=B, 2=C, 3=D, 41 Hdefine BLINK_PORT_NUMBER 42 Hdefine BLINK_PORT_NUMBER 43 Hdefine BLINK_PIN_NUMBER 44 Hdefine BLINK_PIN_NUMBER 44	own board. e GREEN led, C12, active low) f103RB). 4=E, 5=F, 6=0, (1) // PB = 1 (2) // gpio2 = 2 (1)
	45 #define BLINK_GPIOx(_N) 46 #define BLINK_PIN_MASK(_N) 47 #define BLINK_RCC_MASKx(_N)	<pre>((GPI0_TypeDef *)(GPI0A_BASE + (GPI0B_BASE-GPI0A_BASE)*(_N))) (1 << (_N)) (RCC_APB2Periph_GPI0A << (_N))</pre>

图 29、修改代码中的 GPIO 引脚定义为 PB2

11、接下来,就可以编译工程了。右键单击工程名,在弹出的选项中,点击"Build Project",

FTA3	73	- RITNK ON TICKS.
⊳ 🐝 Bir ⊳ 🔊 Inc	New Go Into	
🔺 🔂 sro	Open in New Window	
D 4	Show in Local Terminal	
	🗎 Сору	Ctrl+C
	🛅 Paste	Ctrl+V
	🔀 Delete	Delete
D 🐸 sy:	Remove from Context	Ctrl+Alt+Shift+Down
De 🔁 De	Source	
🔺 🗁 ind	Move	
	Rename	F2
	🔤 Import	
⊳ 🗁 lds	🖆 Export	
	Build Project	
	Clean Project	
1	8 Refresh	F5
× I	Close Project	
2	Close Unrelated Projects	

图 30、编译工程

编译成功后,将会生成 hex 文件与 elf 文件

COTSULT Console Console (ETA321_LED) amm-none-eabi-gcC -mcpu=cortex-m3 -mtnumb -0g -tmessage-length=0 -tsigned-char -trunction-sections -tradat-sections -trreestand arm-none-eabi-get -mcpu=cortex-m3 -mthumb -0g -fmessage-length=0 -fsigned-char -ffunction-sections -fdata-sections -ffreestand arm-none-eabi-size --format=berkeley ETA321_LED.elf ETA321_LED.elf text data bss dec hex filename 5851 176 416 6443 192b ETA321_LED.elf 16:14:24 Build Finished (took 5s.650ms)

图 31、工程编译成功后的信息

三、使用 flashloader 工具烧写 ETA321 应用程序

如果安装好了 Flashloader 工具,这时,可以找到工程目录下编译出来的 HEX 文件,利用 Flashloader 工具将其烧写到 ETA321 中,查看程序运行结果。烧写方法如下:

1、拔动开关到 ETA321 模块上 "3.3V" 端,使 ETA321 处于下载状态,即: ETA321 模块上的红色 LED 灯点亮。用 USB 线连接 ETA321 到 PC 机的 USB 端口。如果是首次进行连接,需要安装 USB 驱动,等一点时间,直到驱动自动安装完成。



图 32、首次安装 USB 驱动

2、从"设备管理器"中,查看 ETA321 模块虚拟的串口编号,即 USB-SERIAL CH340 虚拟串口编号。





3、启动 Flashloader,"Port Name"端口选择 CH340 对应的串口编号,校验选择"none"。

Flash Loader Demonstrator	×
life.augmented	
Select the communication port and set settings, then click next to open connection	
Common for all families	
Data Bits 8 Timeout(s) 10	
Back Next Cancel Close	

图 34、Flashloader 通讯参数

如果确认这里的参数都正确,但是又不能连接到芯片,则需要重新检查一下 CH340 虚 拟串口的"端口设置"属性,并将波特率(位/秒)更改到 115200 即可。

	位/秒(<u>B</u>):	115200		\supset
	数据位(<u>D</u>):	8		•
Ę	奇偶校验(<u>P</u>):	无		•
	停止位(<u>s</u>):	[1		•
	流控制(<u>F</u>):	Æ		-
	高	及(<u>A</u>)	还原默认值	(<u>R</u>)

图 35、ETA321 虚拟串口参数属性

4、点击下一步,Flashloader 会读取到芯片相关的存贮器信息,再下一步,就可以到烧

A HEREINIE

写界面。

X

Flash Load	der Demons	trator				x
		57	lite.augme	nted		
Please, sele	ct your device	in the targe	tlist			
Target	STM32F1_H	ligh-density	_256K		•	
PID (h)	0414					
BID (h)	NA					
Version	2.2					
Flach mannir	1					
Namo	iy Start add	End add	Size			
A Degol	0~ 80000	0~ 80007	0/20 (2K)	68		-
A Paget	0x 80008	0x 80007	0x000 (2K)	ÄÄ		Ŧ
A Page?	0x 80010	0x 80017	0x800 (2K)	ÄÄ		
A Page2	0x 80018	0x 8001F	0x800 (2K)	ÄÄ		
Page4	0x 80020	0x 80027	0x800 (2K)	ĂĂ		
A Page5	0x 80028	0x 8002F	0x800 (2K)	ĂĂ		
Seque6	0x 80030	0x 80037	0×800 (2K)	ĂĂ		
Sequer	0x 80038	0x 8003F	0x800 (2K)	āā		
Seque8	0x 80040	0x 80047	0x800 (2K)	88		
Seque9	0x 80048	0x 8004F	0x800 (2K)	88		
Sequence Page 10	0x 80050	0x 80057	0x800 (2K)	88		
Sequential Apple 1	0x 80058	0x 8005F	0x800 (2K)	88		
🔦 Page12	0x 80060	0x 80067	0x800 (2K)	66		
SPage13	0x 80068	0x 8006F	0x800 (2K)	68		-
Pane14	0v 80070	Ny 80077	0v800 (2K)			
Legend :	🔁 Pro	otected	🔂 UnPi	rotected		
	Back	N	ext	Cancel	Clo	se

图 36、Flashloader 工具读取到的芯片参数

5、选中"Download to device"单选按钮,并将"Verify after download"选择上,即下载完



🔎 Fla	sh Loader Demonstra	ator		
			mented	
C Er	rase			
	 All 	C Sele	ection	
© D	ownload to device - Download from file			
	Erase necessary p	ages (No	Erase 🦰 Glob	al Erase
	(h) 8000000 Optimize (Remove : Apply option bytes	some FFs)	 Jump to the user p Verify after downlo 	rogram
ן ר ט ן	pload from device			
C E	 Enable/Disable Flash pro	otection		
	DISABLE	VRITE PP		
C E	dit option bytes			
	Back	Next	Cancel	Close

图 37、烧写界面

点击"Download from file"右侧的按钮,在弹出的页面中,选择需要烧写的 hex 文件。 需要注意,这里要指定文件类型, Flashloader 软件默认是*.s19,要改为*.hex,否则在文件 夹中看不到相应的 hex 文件。

✓打开	Second de	X
😪 🍚 – 📕 « STM32_XXX 🕨 ET/	A321_LED > Debug >	▼ ↓ 搜索 Debug P
组织 * 新建文件夹		≣ - □ 0
📕 桌面	▲ 名称 ▲	修改日期
	src 📕	2017/10/12 16:14
ARY I DIU	📕 system	2017/10/12 16:14
Ma E	_ ETA321_LED.hex	2017/10/12 16:14
 ■ 视频 ● 图片 □ 文档 ● 音乐 		
🍓 计算机		
👟 OS (C:)		
I Work (D:)		
🛷 TmpBak (G:)	 ✓ ✓ 	•
文件名(N): ET4	A321_LED.hex	hex Files (*.hex) 打开(<u>O</u>) 取消

图 38、选择烧写文件

选择好文件后,点击"打开"按钮,返回烧写页面,再点击"NEXT"便开始烧写。进度条显示绿色,即烧写成功。

By the

Target	STM32F1_High-density_256K
Map file	STM32F1_High-density_256K.STmap
Operation	DOWNLOAD
File name	D:\Project\software\STM32_XXX\ETA321_LED\Debug\ETA321_LE
File size	5.89 KB (6027 bytes)
Status	5.89 KB (6027 bytes) of 5.89 KB (6027 bytes)
Time	00:02
	ownload operation finished successfully

图 39、文件烧写成功

四、利用 J-Link 调试程序代码

首先安装 J-Link 的驱动,在我们提供的软件工具包中也有:JLink_Windows_V620c.exe。 安装完成后,就可以使用 Jlink 进行软件调试了。在调试的时候,ETA321 要处于运行状态, 即:拔动 ETA321 模块上的开关到 S2 端,红色 LED 不亮。

使用 JLink 进行调试之前,先用 JLink GDB Server 工具进行芯片连测试,以确认 Jlink 在正常工作。连接测试流程如下:

1、使用 JLink 的 SWD 信号接口, 连接到 ETA321 的 CN3 插针上, 并给 ETA321 上电, 且 ETA321 处于运行状态。

2、启动 GDB Server 工具



图 40、启动 J-Link GDB Server

3、首先选择器件,GDB 启动起来后,点击"Target Device"右侧的按钮,弹出"Target device settings"页面。在"Manufacturer"中选择"ST",在"Core"中选择"Cortex-M3",然后在下面的列表中,选中"STM32F103RC",最后点击"OK"。

				Teole #	-
Manufacturer	Device	Core	NumCores	Flash size	RAM size
ST	STM32F103R4 (allow opt. bytes)	Cortex-M3	1	16400 Bytes	6 KB
ST	STM32F103R4	Cortex-M3	1	16 KB	6 KB
ST	STM32F103R6 (allow opt. bytes)	Cortex-M3	1	32784 Bytes	10 KB
ST	STM32F103R6	Cortex-M3	1	32 KB	10 KB
ST	STM32F103R8 (allow opt. bytes)	Cortex-M3	1	65552 Bytes	20 KB 🗆
ST	STM32F103R8	Cortex-M3	1	64 KB	20 KB -
ST	STM32F103RB (allow opt. bytes)	Cortex-M3	1	131088 Bytes	20 KB
ST	STM32F103RB	Cortex-M3	1	128 KB	20 KB
ST	STM32F103RC (allow opt. bytes)	Cortex-M3	1	262160 Bytes	48 KB
ST	C STM32F103RC	Cortex-M3	1	256 KB	48 KB
ST	S rivi92F103HD (allow opt. bytes)	Cortex-M3	1	393232 Bytes	64 KB
ST	STM32F103RD	Cortex-M3	1	384 KB	64 KB
ST	STM32F103RE (allow opt. bytes)	Cortex-M3	1	524304 Bytes	64 KB
ST	STM32F103RE	Cortex-M3	1	512 KB	64 KB
ST	STM32F103RF (allow opt. bytes)	Cortex-M3	1	786448 Butes	96 KB
ŝŤ	STM32F103RF	Cortex-M3	1	768 KB	96 KB
		III		1000000	4

图 41、选择目标器件

4、器件选择完以后,还需要配置 GDB Server 连接参数:使用 USB 连接 J-Link, SWD 接口连接目标芯片,1000Khz 速率,最后点击"OK"进行芯片连接。

● USB)	ło	
Target device Unspecified Little endian 💌		
Target interface SWD Speed Auto selection Adaptive clocking	▼ Misc. settings Init registers	
Command line option	conified if Clu/D, and 1000 min	

图 42、配置 GDB 参数

5、连接成功以后, J-Link 项与 CPU 项显示绿色状态,并能读出 CPU 的工作电压。

ile <u>H</u> elp		
GDB Waiting for connection J.Link Connected CPU STM32F103RC	Initial SWD speed 1000 kHz Current SWD speed 1000 kHz 2.98 V Little endiar	Localhost only Stay on top Show log <u>w</u> indow Generate logfile Verify downloac Init regs on starl
J-Link settings file: Target related set Target device: Target interface: Target interface speed: Target endian:	none tings STM32F103RC SWD 1000kHz little	*
Connecting to J-Link I-Link is connected. Tirmware: J-Link V9 comp Hardware: V9.20 Feature(s): RDI, GDB, F1 Checking target voltage. Target voltage: 2.95 V Listening on TCP/IP port Connecting to target 0	oiled Sep 26 2017 17:01:02 ashDL, FlashBP, JFlash, RDDI 2331 ownected to target	Ξ
Jaiting for GDB connecti	on	

图 43、连接成功

到这时,说明 JLINK 的驱动以及硬件连接环境均正是正常,就可以在 eclipse 环境中,利用 Jlink 进行程序调试了。调试方法如下:

- 1、启动 eclipse,打开之前已经编译通过的工程文件 ETA321_LED。
- 2、从菜单栏上选择"RUN"下面的"Debug Configurations"。



图 44、选择 Debug Configurations

3、在左边列表中,选中"GDB SEGGER J-Link Debugging",再点鼠标右键,在弹出的 菜单上,点击"New",在该项目下建立一个新的调试项目。

Create, manage, and run config	jurations	Ŕ
Yue filter text C/C++ Application C/C++ Application C/C++ Attach to Applicatio C/C++ Remote Applicatio C/C++ Remote Applicatio GDB Hardware Debugging GDB OpenOCD Debugging GDB OpenOCD Debugging GDB SEGGER J-Link Debug Launch Group New Launch Group New Launch Group New Filter matched 11 of 11 items	Configure launch settings from this dialog: • Press the 'New' button to create a configuration of the selected type. • Press the 'Duplicate' button to copy the selected configuration. • Press the 'Delete' button to remove the selected configuration. • Press the 'Delete' button to configure filtering options. • Edit or view an existing configuration by selecting it. Configure launch perspective settings from the <u>'Perspectives'</u> preference page.	

图 45、新建调试项目

4、点击新建立的调试项目,展示出右侧的配置页面,这里主要需要配置/检查"Debugger" 与"Startup"这两个配置页面。

在"Startup"配置页面中,去掉如下图中红圈中的两项,并点击"Apply",以确认应用。

Debug Configurations	
Create, manage, and run co	ifigurations
C 🗎 🗶 🖻 🏇 🗸	Name: ETA321_LED Debug
type filter text C/C++ Application C/C++ Attach to Applica C/C++ Postmortem Del C/C++ Remote Applica GDB Hardware Debugg GDB OpenOCD Debugg GDB QEMU Debugging GDB QEMU Debugging GDB SEGGER J-Link Deb C ETA321_LED Debug Launch Group Launch Group	Main Debugger Startup Source Common Initialization Commands Initialization Commands Initial Reset and Halt Type: Low speed: 1000 kHz JTAG/SWD Speed: Auto Adaptive Fixed kHz Enable flash breakpoints Bable semihosting Console routed to: Telnet GDB client Erable SWO CPU freq: 0 Hz. SWO freq: 0 Hz. Port mask: 0x1 Load Symbols and Executable V Load symbols Use project binary: ETA321_LED.elf Use file: Workspace File System Symbols offset (hex): V Load executable Use project binary: ETA321_LED.elf Use file: Workspace File System
← III → Filter matched 12 of 12 items	Re <u>v</u> ert Apply
•	Debug Close
×	图 46、配置 Startup
"Debugger"配置页面	中,需要检查多项参数:
1、检查"Actual exec	utable"参数指向 JLinkGDBServerCL.exe;

- 2、并输入"Device name"为"STM32F103RC";
- 3、 "Endianness" 选择为"Little";
- 4、"Connection"选择为"USB";
- 5、"Interface"选择为"SWD";
- 6、"Initial speed"配置为 1000KHz。

确认参数正确以后,点击"Apply"应用配置参数。最后点击"Debug"进入调试。

reate, manage, and run configurat	ions				5
🗎 🗶 📄 🎲 🔻	Name: ETA321_LE	D Debug			
pe filter text	🗎 Main (🌣 Debug	ger > Startup 5 Source	Common		
C/C++ Application C/C++ Attach to Application	J-Link GDB Server Setup				
	Start the J-Link GDB server locally				
C/C++ Positionern Debugger	Executable:	{jlink_path}/\${jlink_gdbserv	er}	Browse	Variables
 GDB Hardware Debugging GDB OpenOCD Debugging GDB PyOCD Debugging GDB QEMU Debugging CDB QEMU Debugging CDB QEGOTA J Link Debugging ETA321_LED Debug Launch Group Launch Group (Deprecated) 	Actual executable: C:/Program Files (x86)/SEGGER/JLink_V620c//JLinkGDBServerCL.exe				
	(to change it use the global or workspace preferences pages or the project properties page)				
	Device name:	STM32F103RC		Supported	device names
	Endianness:	● Little ◎ Big			
	Connection:		(USB serial or IP name/address)		
	Interface:	SWD ◎ JTAG			
	Initial speed:	🛛 🖉 Auto 💿 Adapti 💿 Fixed	1000 kHz		
	GDB port:	2331			
	SWO port:	2332	Verify downloads Initialize registers on start Z Local host only		
	Telnet port:	2333			
	Log file:				Browse
	out-		a contrat		browse
	Other options:	-singlerun -strict -timeout u	-nogui	tine and GMO	
	Allocate cons	ole for the GDB server	Allocate console for semino	sting and swo	
	GDB Client Setup				
	Executable: \${cross_prefix}gdb\${cross_suffix}				
ter matched 12 of 12 items	-			Revert	Apply
2)				Debug	Clos

图 47、配置 Debugger

7、启动调试以后,程序入口停止在 main()函数,这时可以按 F6 进行单步运行、F5 可 以进入函数内部跟踪运行。

All and a second a

X