

GW1NS-2C MCU IDE

软件参考手册

RN519-1.1,2018-11-21

版权所有©2018 广东高云半导体科技股份有限公司

未经本公司书面许可,任何单位和个人都不得擅自摘抄、复制、翻译本文档内容的部分或全部,并不得以任何形式传播。

免责声明

本文档并未授予任何知识产权的许可,并未以明示或暗示,或以禁止发言或其它方式授予任何知识产权许可。除高云半导体在其产品的销售条款和条件中声明的责任之外,高云半导体 概不承担任何法律或非法律责任。高云半导体对高云半导体产品的销售和/或使用不作任何 明示或暗示的担保,包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知 识产权的侵权责任等,均不作担保。高云半导体对文档中包含的文字、图片及其它内容的准 确性和完整性不承担任何法律或非法律责任,高云半导体保留修改文档中任何内容的权利, 恕不另行通知。高云半导体不承诺对这些文档进行适时的更新。

版本信息

日期	版本	说明
2018/08/21	1.0	初始版本。
2018/11/21	1.1	 ● 优化 Eclipse 安装过程 ● 增加工程实例操作

目录

目录	1
图目录	3
1 ARM KEIL 软件	5
1.1 软件安装	5
1.2 工程模板	5
1.2.1 配置选项	5
1.2.2 工程配置	
1.2.3 工程编译	11
1.2.4 工程烧录	
1.2.5 工程调试	
1.3 参考设计	
2 GNU MCU Eclipse 软件	14
2.1 软件安装	
2.1.1 安装 JDK	
2.1.2 安装 J-LINK	
2.1.3 安装 MCU 器件包	
2.2 软件配置	
2.2.1 导入模板工程	
2.2.2 选择 Properties	
2.2.3 配置环境变量	
2.2.4 配置工具链	
2.2.5 配置烧录工具	
2.2.6 配置 JTAG 模式切换工具	
2.3 工程模板	
2.3.1 工程创建	
2.3.2 工程创建	
2.3.3 工程编译	
10.1.1	

2.3.4 工程烧录	43
2.3.5 工程调试	44
2.4 参考设计	48

图目录

图	1-1 创建工程	6
图	1-2 配置器件	6
图	1-3 配置 ROM 和 RAM	7
图	1-4 配置输出文件格式	8
图	1-5 配置宏定义和头文件路径	8
图	1-6 配置下载选项	9
图	1-7 配置调试选项	10
图	1-8 调试接口类型	10
图	1-9 工程编译	11
图	1-10 烧录	12
图	1-11 JTAG 模式切换命令	12
图	1-12 启动调试	13
图	2-1 GNU MCU Eclipse 安装包目录结构	14
图	2-2 JDK 和 J-LINK 安装目录结构	15
图	2-3 JDK 安装向导	16
图	2-4 选择安装 JDK 可选功能	17
图	2-5 选择 JRE 安装路径	17
图	2-6 完成 JDK 安装	18
冬	2-7 测试 JDK 安装	18
图	2-8 J-LINK License Agreement	19
图	2-9 选择安装路径	20
图	2-10 选择功能组件	20
图	2-11 开始安装向导	21
图	2-12 完成安装	22
图	2-13 选择 Preferences	23
图	2-14 指定 Packages 路径	24
图	2-15 更新器件包列表	25
RN519-1	.1	3(44)

图	2-16 导入模板工程	26
图	2-17 选择 Properties	27
图	2-18 配置环境变量	28
图	2-19 配置工具链	28
图	2-20 配置烧录工具	29
图	2-21 配置 JTAG 模式切换工具	30
图	2-22 交叉编译配置	30
图	2-23 新建工程	31
图	2-24 选择工程类型配置	32
图	2-25 选择工具链	33
图	2-26 工程结构	34
图	2-27 选择 Properties	35
图	2-28 配置 Cross ARM GNU Assembler Preprocessor	36
图	2-29 配置 Cross ARM GNU Assembler Includes	37
图 图	2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor	37 38
图 图 图	 2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor 2-31 配置 Cross ARM C Compiler Includes 	37 38 39
图 图 图 图 图	 2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor 2-31 配置 Cross ARM C Compiler Includes 2-32 配置 Cross ARM C Linker 	37 38 39 40
图 图 图 图 图 图 图	 2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor 2-31 配置 Cross ARM C Compiler Includes 2-32 配置 Cross ARM C Linker 2-33 配置 Cross ARM GNU Create Flash Image. 	37 38 39 40 41
图 图 图 图 图 图 图 图 图 图	 2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor 2-31 配置 Cross ARM C Compiler Includes 2-32 配置 Cross ARM C Linker 2-33 配置 Cross ARM GNU Create Flash Image 2-34 配置 Devices 	37 38 39 40 41 42
图 图 图 图 图 图 图 图 图 图 图	2-29 配置 Cross ARM GNU Assembler Includes	 37 38 39 40 41 42 43
图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图	2-29 配置 Cross ARM GNU Assembler Includes	 37 38 39 40 41 42 43 43
图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图	2-29 配置 Cross ARM GNU Assembler Includes	 37 38 39 40 41 42 43 43 44
图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图	2-29 配置 Cross ARM GNU Assembler Includes	 37 38 39 40 41 42 43 43 44 45
图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图	2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor 2-31 配置 Cross ARM C Compiler Includes 2-32 配置 Cross ARM C Linker 2-33 配置 Cross ARM GNU Create Flash Image. 2-34 配置 Devices 2-35 编译工程 2-36 工程烧录 2-37 建立调试配置选项 2-38 配置 Image 路径 2-39 配置 Debugger.	 37 38 39 40 41 42 43 43 44 45 46
图图图图图图图图图图图	2-29 配置 Cross ARM GNU Assembler Includes 2-30 配置 Cross ARM C Compiler Preprocessor 2-31 配置 Cross ARM C Compiler Includes 2-32 配置 Cross ARM C Linker 2-33 配置 Cross ARM GNU Create Flash Image 2-34 配置 Devices 2-35 编译工程 2-36 工程烧录 2-37 建立调试配置选项 2-38 配置 Image 路径 2-39 配置 Debugger 2-40 JTAG 接口切换	 37 38 39 40 41 42 43 43 44 45 46 47

1 ARM KEIL 软件

1.1 软件安装

请参考 ARM 提供的<u>《MDK Getting Started》</u>和<u>《uVision User's Guide》</u> (可通过以下路径获取该手册: GW1NS-2C_package\release\rt...)。

1.2 工程模板

使用 ARM KEIL 软件进行嵌入式软件开发,需要创建工程、配置工程选项、编码、编译、烧录和调试。

1.2.1 配置选项

工程创建

打开 ARM KEIL 软件,选择菜单栏 Project 中 New uVision Project..., 创建工程,如图 1-1 所示。 图 1-1 创建工程



1.2.2 工程配置

配置器件

GW1NS-2C MCU 为 ARM Cortex-M3 内核,所以器件选择 ARM Cortex-M3 的"ARMCM3",如图 1-2 所示。

图 1-2 配置器件

Device Target Output Listing Vser	C/C++ Asm Linker Debug Vtilities
Software Packs Vendor: ARM Device: ARMCM3 Toolset: ARM Search:	Software Pack Pack: ARM.CMSIS.5.2.0 URL: <u>http://www.keil.com/pack/</u>
ARM ARM Cortex M0 ARM Cortex M0 ARM Cortex M23 ARM Cortex M3 ARM Cortex M3 ARM Cortex M3 ARM Cortex M33 ARM Cortex M34 ARM Cortex M	The Cortex-M3 processor is an entry-level 32-bit ARM Cortex processor designed for a broad range of embedded applications. It offers significant benefits to developers, including: - simple, easy-to-use programmers model - highly efficient ultra-low power operation - excellent code density - deterministic, high-performance interrupt handling - upward compatibility with the rest of the Cortex-M processor family.
OK	Cancel Defaults Help

配置 ROM 和 RAM

配置 ROM 和 RAM 的起始地址和容量大小。

GW1NS-2C Flash-Rom 起始地址为 0x00000000, 容量为 128K Byte。

GW1NS-2C Sram 起始地址为 0x20000000,容量为 8K Byte, 如图 1-3 所示。

图 1-3 配置 ROM 和 RAM

Device Target ()1tput List	ing Vser	C/C++ A	sm]	Linker I	ebug Util	ities	
ARM DS_CM3				Code C	Generation			
		Xtal (MHz):	2.0	ARM	Compiler:	Use default	compiler version	n 5 🔻
Operating system:	None		-					
System Viewer File:				🗆 🗆 U	se Cross-N	lodule Optimiza	tion	
DS_CM3.svd				🗆 🗆 U	se MicroLl	в Г	Big Endian	
Use Custom File	e		_					
Read/Only Memo	ny Areas ——			Read/	Write Mem	ory Areas		
default off-chip	Start	Size	Startup	default	off-chip	Start	Size	NoInit
ROM1:			0		RAM1:			
ROM2:			0		RAM2:			
ROM3:			- o		RAM3:			
on-chip			_		on-chip			
IROM1:	0x0	0x20000	œ		IRAM1:	0×20000000	0x2000	
IROM2:			0		IRAM2:			
						· · · · · · · · · · · · · · · · · · ·		
		OK	Cano	rel	Def	aults		Help

配置输出文件格式

高云 Programmer 支持 BIN 二进制文件烧录格式,所以配置输出文件格式为 BIN 格式。

User 命令行选项中转换*.axf 文件为*.bin 文件,如图 1-4 所示。

图 1-4 配置输出文件格式

Device Target Output Listin	g User C/C++ Asm Linker Debug Ut	ilit	ies	[
Command Items	User Command		Stop on Exi	S	
Before Compile C/C++ File					E
Run #1		2	Not Specified		
— Run #2		2	Not Specified		
Before Build/Rebuild					
Run #1		2	Not Specified		
Run #2		2	Not Specified		
E-After Build/Rebuild					
Run #1	C:\Keil_v5\ARM\ARMCC\bin\fromelf.exebin -o	.\Ob	jects\led.bin .\(Dbjects\led.	axf
🗌 🗌 Run #2			Not Specified		
Run 'Atter-Build' Conditionally					
✓ Beep When Complete	☐ Start Debugging				
	OK Cancel Defaults		н	elp	

命令格式为:

*:\Keil_v5\ARM\ARMCC\bin\fromelf.exe --bin

-o ./Objects/*.bin ./Objects/*.axf

配置宏定义和头文件路径

配置宏定义"USE_STDPERIPH_DRIVER",用来调用标准外设。 配置头文件路径,编译过程中用来调用头文件。配置如图 1-5 所示。 图 1-5 配置宏定义和头文件路径

Define: USE_STDPERIPH_DRIVER	3	
Language / Code Generation Execute-only Code Optimization: Level 0 (-00) Optimize for Time Split Load and Store Multiple One ELF Section per Function	Strict ANSI C Frum Container always int Plain Char is Signed Read-Only Position Independent Read-Write Position Independent	Wamings: All Wamings Thumb Mode No Auto Includes V C99 Mode
Include	is:\STARTUP:\SYSTEM:\USER 00 -apcs=interwork -split_sections -1/CORE 1/USER	-1/PERIPH/Includes -1

配置烧录工具

配置 Flash 下载工具为外部工具,使用高云 Programmer,如图 1-6 所示。

图 1-6 配置下载选项

Device Target Output Listing Vser C/C++	Asm Linker Debug Utilities
Configure Flash Menu Command	
C Use Target Driver for Flash Programming	Use Debug Driver
Use Debug Driver	Settings 🔽 Update Target before Debugging
Init File:	Edit
Use External Tool for Flash Programming	
Command: D:\Gowin\1.8\programmer2\bin\program	mer_cli.exe
Arguments:device GW1NS-2C -run 22 f D:\user-b	ak\Users\root\Desktop\ed\PROJECT\Objects\ed.bin
Run Independent	
Configure Image File Processing (FCARM):	
Output File:	Add Output File to Group:
	STARTUP
Image Files Root Folder:	Generate Listing
OK	ancel Defaults Help

Command 为 Programmer 路径。

Arguments 为 Programmer 命令选项,如--device GW1NS-2C --run 22 -f *.bin。

配置调试选项

如果选择使用 U-LINK 仿真器,则 Debug 选项配置为"ULNK2/ME Cortex Debugger"。

如果选择使用 J-LINK 仿真器,则 Debug 选项配置为"J-LINK/J-TRACE Cortex"。如图 1-7 所示。

图 1-7 配置调试选项

Device Target Output Listing User C/C++ A	Asm Linke <mark>r Debug V</mark> ilities
O Use Simulator with restrictions Settings	Use: ULINK2/ME Contex Debugger Settings
Limit Speed to Real-Time	ULINK2/ME Cortex Debugger
✓ Load Application at Startup ✓ Run to main() Initialization File:	CMSIS-DAP Debugger CMSIS-DAP Debugger main() Initials
Edit	ST-Link Debugger
Restore Debug Session Settings	Restore Stellaris ICDI
I Breakpoints I Toolbox	Bre SiLabs UDA Debugger
✓ Watch Windows & Performance Analyzer	Watch windows
Memory Display Visiting System Viewer	Very Memory Display Very System Viewer
CPU DLL: Parameter:	Driver DLL: Parameter:
SARMCM3.DLL -MPU	SARMCM3.DLL -MPU
Dialog DLL: Parameter:	Dialog DLL: Parameter:
DCM.DLL -pCM3	TCM.DLL pCM3
Manage Component Vie	ewer Description Files
OK	Cel Defaults Help

调试接口类型配置为 JTAG, 如图 1-8 所示。

图 1-8 调试接口类型

Cortex-M Target Driver Setup	-		loss here mill	×
Debug Trace Flash Download				
ULINK USB - JTAG/SW Adapter		Device Chain		
Serial No: Any		IDCODE	Device Name	IR len Move
ULINK Version: ULINK2	TDO	⊙ 0x4BA00477	ARM CoreSight JTAG-DP	4 Up
Device Family: Cortex-M	TDI			Down
Firmware Version: V2.03	• Au	tomatic Detection	ID CODE:	
SWJ Port: JTAG -	О Ма	anual Configuration	Device Name:	
Max Clock: 1MHz 💌	Add	d Delete Up	odate IR len:	AP: 0x00
Debug Connect & Reset Options Connect: Normal Reset Reset after Connect Log Debug Accesses	:: Autodet	tect 🔽	Cache Options ──	nload Options (erify Code Download)ownload to Flash
L		ок с	ancel	Help

1.2.3 工程编译

完成编码和工程配置后,编译生成 BIN 格式镜像文件,如图 1-9 所示。 图 1-9 工程编译

<pre>Project led Id = Startup</pre>	La La pe La 144 led	
Paper Hel Hel Attribute Comparison Comparison () 2014-2018 Gevin Semiconductor Technology Co., itsd. PROPHRAL File Media Comparison Com	1 10	and a second s
<pre>statup statup statup statup soft soft soft soft soft soft soft soft</pre>	Project: led	1
<pre>G STATUP G COR G COR G COR G STIM G STI</pre>	😥 led	2 💷 /*
<pre>CODE CODE CODE CODE CODE CODE CODE CODE</pre>	E STARTUP	3 • •••••••••••••••••••••••••••••••••••
<pre>> STATA > STATA > PROPHATING to a state of the state</pre>	E CORE	4
<pre></pre>	SYSTEM	S Copyright (C) 2014-2018 Gowin Semiconductor Technology Co., Ltd.
<pre></pre>		
<pre></pre>	PERIPHERAL	8 Sauthor Fundadid Development Team
<pre> #</pre>	E G USER	9 * Supraino VI.C.O
<pre># make: * @citer Main program body. *// Toludes *// To</pre>	🗟 🛄 gw1ns2k_it.c	10 * #date 2018-5-1 09:00:00
<pre></pre>	III i main.c	11 * Sbrief Main program body.
<pre> */</pre>		12
<pre>14 / Includes // Includes // Includes // Includes // Includes // Include // In</pre>		13 */
<pre>18 /* Includes</pre>		14
<pre> # fielded = Typingk.m # # fielded = Typingk.m # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # # #</pre>		15 /* Includes*/
<pre> # Incluse 40430.15 # Incluse 40</pre>		16 finclude "gwlms2k.h"
<pre> # Incluse = Tpuint()/20% is = #333000 #2 for(focust = 0; focust-); id focust = 0; focust =</pre>		17 #include <#tdio.h>
<pre> function Cutting.in function functio</pre>		18 #include "guinszk gplo.n"
<pre>21 void bear(_10 vints2 t nCount)//38% is = 8338000 22 { tor; noous i= 0; nCount); i tor; noous i= 0; nCount); tor; noous i= 0; ncount]; tor; noous i= 0; n</pre>		13 Finction Storing by
<pre>22 E1(22 E1(23 E1(24 E1(24 E1(24 E1(25 E</pre>		21 word Delaw(IO winr32 r pCoupr)//25W is = 2333000
<pre>22 True: nount := 0: nCount); 23 / 24 / 25 / 25 / 26 inc.main(void) 27 / 28 / 28 / 29 / 29 / 29 / 20 /</pre>		
<pre>24 1 24 24 25 25 26 27 26 27 27 27 27 27 27 27 27 27 27 27 27 27</pre>		<pre>23 for(; nCount != 0; nCount);</pre>
<pre>26 L train(void) 27 [{ 26 L train(void) 27 [{ 26 J train(void) 27 [{ 26 J train(void) 27 [{ 27 J generaliz(1); 28 J voile(1) 29 J generaliz(1); 29 J generaliz(1); 29 J (20 J generaliz(1); 20 20 J g</pre>		24 3
<pre>26 int main(rodd) 27 [i ypsemlint(); 30 gPtco-southstr = oxffffff; 31 wile(1) 31 gPtco-southstr = oxffffff; 33 gPtco-southstr = oxffffff; 34 gPtco-southstr = oxffffff; 35 gPtco-southstr = oxffffff; 36 gPtco-southstr = oxffffff; 37 gPtco-southstr = oxffffff; 38 gPtco-southstr = oxffffff; 39 gPtco-southstr = oxfffff; 39 gPtco-southstr = oxffff; 39 gPtco-southstr = oxffff; 39 gPtco-southstr = oxffff; 39 gPtco-southstr = oxffff; 39 gPtco-southst = oxffff; 39 gPtco-southst = oxffff; 39 gPtco-southst = oxffff; 39 gPtco-southst = oxfff; 39 gPtco-s</pre>		25
<pre>27 [1 28 39 cpresminit(); 30 cproc-contrast = 0xfffffff; 31 cmle(); 33 cproc_sets(cproc,oproc_sets_cproc_</pre>		26 int main (vold)
<pre>3 Systemilut(); 20 Systemilut(); 21 Systemilut(); 32 (33 GPIO>-SOTISHIT = OxfIFIFIF; 34 GPIO>-SotIst(OPIO,OPIO_Pin_2)(OPIO_Pin_3); 35 (/OPIO>-SATAOT = Oxide: 37 Delay(33300); 38 GPIO_BeenExi(OPIO,OPIO_Pin_2)(OPIO_Pin_3); 39 (OPIO>-SatAOT = Oxide: 30 Delay(33300); 39 (OPIO>-SatAOT = Oxide: 30 Delay(33300); 30 (OPIO_BeenExi(OPIO,OPIO_Pin_2)(OPIO_Pin_3); 31 (OPIO_SeenExi(OPIO,OPIO_Pin_3); 32 (OPIO=BeenExi(OPIO,OPIO_Pin_3); 33 (OPIO_BeenExi(OPIO,OPIO_Pin_3); 34 (OPIO=BeenExi(OPIO,OPIO_Pin_3); 35 (OPIO_BeenExi(OPIO,OPIO_Pin_3); 36 (OPIO_BeenExi(OPIO,OPIO_Pin_3); 37 (OPIO=OxidatOT = Oxide: 38 (OPIO_BeenExi(OPIO,OPIO_Pin_3); 39 (OPIO=BeenExi(OPIO,OPIO_Pin_3); 39 (OPIO=BeenExi(OPIO,OPIO_Pin_3); 39 (OPIO=BeenExi(OPIOO,OPIO_Pin_3); 30 (OPIO_BeenExi(OPIOO,OPIO_Pin_3); 30 (OPIO_BeenExi(OPIOO,OPIO_Pin_3); 31 (OPIO_OXIDE = Oxide: 32 (OPIO=BeenExi(OPIOO); 33 (OPIO_PIN_0); 34 (OPIO=BeenExi(OPIOO,OPIO_Pin_3); 35 (OPIO_PIN_0); 36 (OPIO_PIN_0); 37 (OPIO=BeenExi(OPIOO); 38 (OPIO_PIN_0); 39 (OPIO=BeenExi(OPIOO,OPIO_PIN_0); 39 (OPIO_PIN_0); 39 (OPIO_PIN_0); 30 (OPIO_PIN_0); 30 (OPIO_PIN_0); 30 (OPIO_PIN_0); 31 (OPIO_PIN_0); 32 (OPIO_PIN_0); 33 (OPIO_PIN_0); 34 (OPIO_PIN_0); 35 (OPIO_PIN_0); 35 (OPIO_PIN_0); 36 (OPIO_PIN_0); 37 (OPIO_PIN_0); 38 (OPIO_PIN_0); 39 (OPIO_PIN_0); 39 (OPIO_PIN_0); 39 (OPIO_PIN_0); 39 (OPIO_PIN_0); 30 (OPIO_PIN_0); 30 (OPIO_PIN_0); 30 (OPIO_PIN_0); 30 (OPIO_PIN_0); 30 (OPIO_PIN_0); 31 (OPIO_PIN_0); 32 (OPIO_PIN_0); 33 (OPIO_PIN_0); 34 (OPIO_PIN_0); 35 (OPIO_PIN_0); 36 (OPIO_PIN_0); 37 (OPIO_PIN_0); 39 (OPIO_PIN_0); 39 (OPIO_PIN_0); 30 (OPIO_PIN_0</pre>		27 🖓 (
0 0		28 SystemInit();
<pre>31</pre>		29
0 0		30 GPIO-SOLENSEI = OXIIIIIIII
33 0700 Setäst:(0700,0700 Pin_1)(0F00 Pin_2)(0F10 Pin_3); 34 0700 Pinestik:(0700,0700 Pin_0)(0F10 Pin_2)(0F10 Pin_3); 35 //07100-SATAOUT = SALe2 36 0700 Pinestik:(07100,0700 Pin_0)(0F10 Pin_2)(0F10 Pin_3); 37 0710 Pinestik:(07100,0700 Pin_0)(0F10 Pin_2)(0F10 Pin_3); 38 0710 Pinestik:(07100,0700 Pin_0)(0F10 Pin_2)(0F10 Pin_3); 39 0710 Pinestik:(07100,0700 Pin_0)(0F10 Pin_0)(0F10 Pin_3); 39 0710 Pinestik:(07100,0710 Pin_0)(0F10 Pin_0)(0F10 Pin_3); 30 0710 Pinestik:(07100,0710 Pin_0)(0F10 Pin_0)(0F10 Pin_3); 31 0710 Pinestik:(07100,0710 Pin_0); 32 0710 Pinestik:(07100,0710 Pin_0); 33 0710 Pinestik:(07100,0710 Pin_0); 34 0710 Pinestik:(07100,0710 Pin_0); 35 070 Pinestik:(07100,0710 Pin_0); 36 070 Pin_0 Pin_0); 37 070 Pinestik:(07100 Pin_0); 38 070 Pin_0 Pin_0); 39 070 Pinestik:(07100 Pin_0); 39 070 Pinestik:(07100 Pin_0); 39 070 Pinestik:(07100 Pin_0); 39 070 Pinestik:(071000 Pin_0); 39 <		31 WRAP(1)
<pre>34</pre>		38 GPTO SerBit (GPTO) GPTO Pin 1 GPTO Pin 2 GPTO Pin 3).
<pre>35 //0700->balay(35300); 36 balay(35300); 37 dFL0_betBalay(35300); 38 dFL0_betBalay(35300); 39 dFL0_betBalay(35300); 40 balay(353000); 41 dFL0_betBalay(353000); 42 dFL0_betBalay(353000); 42 dFL0_betBalay(353000); 43 dFL0_betBalay(353000); 44 dFL0_betBalay(353000); 45 dFL0_betBalay(353000); 46 dFL0_betBalay(353000); 47 dFL0_betBalay(353000); 48 dFL0_betBalay(3530000); 48 dFL0_betBalay(353000); 48 dFL0_betBalay(353000); 48 dFL0_betBalay(353000); 48 dFL0_betBalay(3530000); 48 dFL0_betBalay(3530000)</pre>		34 GPIO RemetBat (GPIOD, GPIO Pin 0);
36 Delay(#333000); 37 GFOLSKIE(#FOLOPTO_PIn_0)(#FIO_PIn_2)(#FIO_PIn_3); 38 GFIO_RearCBL(#FIO_OFTO_PIN_1); 39 (GFIO_RearCBL(#FIO_OFTO_PIN_1); 40 Delay(#333000); 41 GFIO_RearCBL(#FIO_OFTO_PIN_2); 42 GFIO_RearCBL(#FIO_OFTO_PIN_2); 43 GFIO_RearCBL(#FIO_OFTO_PIN_2); 44 Delay(#333000); 44 Delay(#333000); 44 Delay(#333000); 45 Delay(#333000); 46 Delay(#333000); 46 Delay(#333000); 47 Delay(#333000); 48 Delay(#333000); 48 Delay(#333000); 49 Delay(#333000); 40 Delay(#333000); 40 Delay(#333000); 40 Delay(#333000); 41 Delay(#333000); 42 Delay(#333000); 44 Delay(#333000); 44 Delay(#333000); 45 Delay(#333000); 46 Delay(#333000); 47 Delay(#333000); 48 Delay(#333000); 48 Delay(#333000); 49 Delay(#333000); 40 Delay(#33000); 40 Delay(#33000); 40 Delay(#33000); 40 Delay(#33000);		35 //GPIOO->DATAOUT = 0xle:
37 GPIC_SetBit (GPICO_GPIC_pin_2)(GPIC_pin_2)(GPIC_pin_3); 38 GPIC_BesetBit (GPICO_GPIC_pin_2)(GPIC_pin_2)(GPIC_pin_3); 39 //GPICO-SARXICGPIC_GPIL_); 38 //GPICO-SARXICGPIC_GPIL_); 39 //GPICO-SARXICGPIC_GPIL_); 30 //GPICO-SARXICGPIC_GPIL_); 31 GPIC_GPILE_GPIC_GPIL_); 42 GPIC_GPILE_GPIL_GPIC_GPIL_); 43 GPIC_GPILE_GPIC_GPIL_); 44 Feller/Althoutort = Sab; 44 Feller/Althoutort = Sab;		36 Delay(8333000);
38 GFIC_DRevetBit(GFIC.0)(pin_1); 39 //GFIC-ONDATOT = 0xit; 40 Delay(sBB000); 41 GFIC_DRevetBit(GFIC.0)(FIC_DIn_0)(GFIC_DIn_0); 43 GFIC_DRevetBit(GFIC.0)(FIC_DIn_0)(GFIC_DIn_0); 44 GFIC_DRevetBit(GFIC.0)(FIC_DIN_0); 44 Matar/SINDON;		37 GPIO_SetBit (GPIO0, GPIO_Pin_0) GPIO_Pin_2) GPIO_Pin_3);
<pre>38 //0700-56XA00T = 048: 00 balay(183000);0,0702 fm_](0FD0_Fin_0)GFT0_Fin_3); 42 gFT0_Fin_0(FT0_Fin_0); 43 gFT0_Fin_0(FT0_Fin_0); 44 gFT0_Fin_0(FT0_Fin_0); 44 gFT0_Fin_0(FT0_Fin_0); 45 gFT0_Fin_0(FT0_Fin_0); 46 gFT0_Fin_0(FT0_Fin_0); 47 gFT0_Fin_0(FT0_Fin_0); 48 gFT0_Fin_0(FT0_Fin_0); 48 gFT0_Fin_0(FT0_Fin_0); 49 gFT0_Fin_0(FT0_Fin_0); 49 gFT0_Fin_0(FT0_Fin_0); 40 gFT0_Fin_0(FT0_Fin_0); 40 gFT0_Fin_0(FT0_Fin_0); 40 gFT0_Fin_0(FT0_Fin_0); 40 gFT0_Fin_0(FT0_Fin_0); 40 gFT0_Fin_0(FT0_Fin_0); 41 gFT0_Fin_0(FT0_Fin_0); 42 gFT0_Fin_0(FT0_Fin_0); 43 gFT0_Fin_0(FT0_Fin_0); 44 gFT0_Fin_0(FT0_Fin_0); 45 gFT0_Fin_0(FT0_Fin_0); 46 gFT0_Fin_0(FT0_Fin_0); 47 gFT0_Fin_0(FT0_Fin_0); 48 gFT0_Fin_0(FT0_Fin_0); 49 gFT0_Fin_0(FT0_Fin_0); 49 gFT0_Fin_0(FT0_Fin_0); 49 gFT0_Fin_0(FT0_Fin_0); 40 gFT0_Fin_0(FT0</pre>		<pre>38 GPIO_ResetBit(GPIO0,GPIO_Pin_1);</pre>
0 Delay(#33300); 41 GTO_Status (GTOLO, GTO_D'En_1)(GTO_PEn_0)(GTO_PEn_3); 43 GTO_Bestellat(GTOLO, GTO_D'En_2); 44 (GTO_Status (GTOLO, GTO_D'En_2); 44 Main(GTO_Status (GTOLO, GTO_D'En_2); 44 Main(GTO_Status (GTOLO, GTO_D'En_2);		39 //GPIOD->DATAOUT = 0xd;
41 GF0[_SetBit(GF00,GF0_Fin_)); 42 GF0[_SetBit(GF00,GF0_Fin_)); 43 //GF10=SetBit(GF00,GF0_Fin_)); 44 Delter(GF0,GF0_Fin_); 45 Delter(GF00,GF0_Fin_); 46 Delter(GF00,GF0_Fin_); 46 Delter(GF00,GF0); 47 Delter(GF00,GF0); 48 Delter(GF00,GF		40 Delay(8333000);
		41 GFIO SetBit (GFIO0, GFIO Pin 1) GFIO Pin 0) GFIO Pin 3);
1 33 7/04100-00048001 = 004007 0 0 0 0 0 0 0 0 0 0 0 0		42 GPIO ResetBit (GPIO, GPIO Pin 2);
The Ofen Ofen Ofen (Ofen)		43 //GFIO-SDRIADUT = UXID:
doud .	🚯 80 () Fu 0. Te	
n par	utput	

1.2.4 工程烧录

完成工程编译和配置高云 Programmer 软件为烧录工具后,选择 Download 完成烧录,如图 1-10 所示。

图 1-10 烧录



1.2.5 工程调试

完成 MCU 镜像文件烧录后,如果用户设计出现问题,可以连接 U-LINK 或 J-LINK 仿真器使用调试功能。

JTAG 模式切换

如图 1-11 所示,使用高云 Programmer 命令行手动将 JTAG 模式由下载 模式切换到调试模式,命令为:

programmer_cli.exe --device GW1NS-2C --run 23

图 1-11 JTAG 模式切换命令



JTAG 接口切换

如果使用开发板"DK-EVAL-GW1NS V1.1", 需要手动将 JTAG 接口 TMS、 TCK、TDI 和 TDO 跳线帽由 FDTI 烧录切换到 ARM 烧录,即由(1,3)端口切 换到(3,4)端口。

启动调试

连接 U-LINK 或 J-LINK 仿真器,选择工具栏 Debug 按钮,开始调试, 如图 1-12 所示。

图 1-12 启动调试



1.3 参考设计

高云提供基于 ARM KEIL 软件的参考设计:

Gowin GW1NS-2C_MCU_PACK\Gowin_GW1NS-2C_MCU_RefDesign\ MCU_RefDesign\Keil

2GNU MCU Eclipse 软件

2.1 软件安装

高云半导体提供 GNU MCU Eclipse 安装包。

获取 GNU MCU Eclipse 安装包,解压到本地 D 盘根目录下,如图 2-1 所示。

图 2-1 GNU MCU Eclipse 安装包目录结构

					83
€	≪ 软件 (D:) ▶ eclipse_tools ▶	▼ 4 9 搜索 ecl	lipse_tools		Q
组织 ▼	包含到库中 ▼ 共享 ▼ 刻录	新建文件夹		≡ ▼	?
☆ 收藏夹	名称	修改日期	类型	大小	
	퉬 doc	2018/11/8 10:37	文件夹		
库	퉬 eclipse	2018/11/8 0:41	文件夹		
	퉬 eclipse_prj	2018/11/7 16:39	文件夹		
■ 计算机	퉬 eclipse_work	2018/11/7 19:41	文件夹		
	퉬 installer	2018/11/7 20:50	文件夹		
G 网络	퉬 packages	2018/11/7 23:09	文件夹		
- T 135H	퉬 tools	2018/11/7 20:42	文件夹		
1. 7	7 个对象				

选择 installer 目录, 获取 JDK 安装文件和 J-LINK 仿真器驱动安装文件, 如图 2-2 所示。

图 2-2 JDK 和 J-LINK 安装目录结构



2.1.1 安装 JDK

选择 JDK 安装文件

选择安装文件 jdk-8u192-windows-x64.exe, 双击打开, 如图 2-3 所示。

图 2-3 JDK 安装向导

岁 Java SE Development Kit 8 Update 192 (64-bit) - 安装程序 🛛 🔀
欢迎使用 Java SE 开发工具包 8 Update 192 的安装向导
本向导将指导您完成 Java SE 开发工具包 8 Update 192 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

选择要安装的可选功能和安装目录

选择要安装的可选功能和安装目录,如图 2-4 所示,默认安装路径为 C:\Program Files\Java\jdk1.8.0_192。

图 2-4 选择安装 JDK 可选功能

👸 Java SE Development Kit 8 Update 192 (64-bit) -	定制安装
从下面的列表中选择要安装的可选功能。您可以在安装/ 实用程序更改所选择的功能	后使用控制面板中的"添加/删除程序"
 □・ 开发工具 □・ 源代码 □・ 公共 JRE 	Java SE Development Kit 8 Update 192 (64-bit), 包括 JavaFX SDK, 一个专用 JRE 以及 Java Mission Control 工具套 件。它要求硬盘驱动器上有 180MB 空间。
安裝到: C:\Program Files\Java\jdk1.8.0_192\	更改(C)
<上一步(B)	下一步(N) > 取消

选择 JRE 的安装目录

选择 JRE 的安装目录,如图 2-5 所示,默认路径为 C:\Program Files\Java\jre1.8.0_192。

图 2-5 选择 JRE 安装路径

Java 安装 - 目标文件夹		
	目标文件夹	
单击 "更改" 以将 Java 安装到其他文件夹。		
安装到: C:\Program Files\Java\jre1.8.0_192		更改(C)

完成 JDK 安装

如图 2-6 所示, JDK 安装成功,选择关闭即可。

图 2-6 完成 JDK 安装

闄 Java SE Development Kit 8 Update 192 (64-bit) - 完成	×
Java SE Development Kit 8 Update 192 (64-bit) 已成功安装	
单击"后续步骤"访问教程, API 文档, 开发人员指南, 发布说明及更多内容, 帮助您 开始使用 JDK。	
后续步骤(N)	
关闭(C)	

测试 JDK 安装

本地选择开始菜单,选择打开 cmd 命令行,输入命令"java-version",如果可以输出 Jave 版本信息,则安装成功,如图 2-7 所示。

图 2-7 测试 JDK 安装



2.1.2 安装 J-LINK

选择安装文件

选择安装文件 Setup_JLink_V490.exe,双击打开,如图 2-8 所示,确认 License Agreement。

如果本地已安装 J-LINK,不需要再次安装 J-LINK,只需在 Eclipse 中设置本地已安装的 J-LINK 的路径。

2-8 J-LINK License Agreement



选择要安装目录

选择要安装的目录,如图 2-9 所示,默认路径为 C:\Program Files (x86)\SEGGER\JLink_V490。

图 2-9 选择安装路径

😼 Choose Destination I	Location	X
	Setup will install J-Link V4.90 in the following folder. To install into a different folder, click Browse, and select another folder. You can choose not to install J-Link V4.90 by clicking Cance to exit Setup. Destination Folder C:\\SEGGER\JLink_V490	
	< <u>B</u> ack <u>N</u> ext > Cancel	

选择要安装的功能组件

选择要安装的功能组件,如图 2-10 所示。

图 2-10 选择功能组件

🔏 Choose options		83
	Choose optional components that should be installed: Install USB Driver for J-Link Install USB Driver for J-Link with Virtual COM Port Choose options for creating shortcuts I create entry in start menu Add shortcuts to desktop	
	< <u>B</u> ack <u>N</u> ext > Can	cel

选择开始安装

开始安装,如图 2-11 所示。

图 2-11 开始安装向导



完成安装

完成安装,如图 2-12 所示。

图 2-12 完成安装

😼 Installation Complete		83
	J-Link V4.90 has been successfully installed. USB Driver for J-Link installed USB Driver for J-Link with Virtual COM Port installed Press the Finish button to exit this installation.	
	< <u>B</u> ack <u>Finish</u> > Cano	el 🛛

2.1.3 安装 MCU 器件包

GNU MCU Eclipse 安装包中提供 MCU 器件包 packages,如图 2-1 所示, packages 为 MCU 器件包。

指定 MCU 器件包路径

选择菜单栏 Window 中的 Preferences,如图 2-13 所示。

eclipse_work - C/C++ - Eclipse					
File Edit Source R	efactor Navigate Search	Project Run W	indow Help		
) Run [□] ≪ ▼ 副 □ ≈ & * ■ ▼ 29 0∋	gpio Debug	New Window Editor > Appearance > Show View > Perspective >		
Pr ☆ □ □ □ ☆ ☞ ▽ ■ ☞ gpio			Navigation Preferences		
 Binaries Includes CORE CORE Core Flash Flash FRIPHER STARTUP SYSTEM USER 	Console & Problem CDT Build Console [gpio] Invoking: cross Akm c C arm-none-eabi-gcc -mcpu Finished building:/C Building target: gpio.e Invoking: Cross ARM C L Invoking: Cross ARM C L	ns compiler J=cortex-m3 -mth CORE/core_cm3.c elf Linker Linker Linker	umb 00 fmocroso		

图 2-13 选择 Preferences

选择 Preferences 中的 C/C++选项卡下的 Packages 选项,指定 Packages folder 和 Macro name,如图 2-14 所示。

 ▷ General ▲ C/C++ Appearance Arduino ▷ Autotools ▷ Build Code Analysis 	The location where package Packages folder: D:\eclipse Macro name: packs_pat	es are stored. e_tools\packages Browse
 Code Style Debug Docker Container Launch Editor File Types Indexer Language Mappings LLVM New C/C++ Project Wizard Packages Repositories Profiling 		Restore Defaults Apply

图 2-14 指定 Packages 路径

在线更新 MCU 器件列表

选择工具栏中的 Packs 按钮,在线更新 MCU 器件包列表,如图 2-14 所示。因为 GNU MCU Eclipse 安装包已经提供 ARM.CMSIS.5.3.0 MCU 器件包,所以图 2-15 中显示该器件包已安装。

eclinse work - Packs	- Eclinse			
File Edit Navigate S	earch Project Run Window H	Help		
* • *	Debug 🗸 🖻 gpio De	ebug	~ ‡	
📬 🖛 📙 🕼 🚍	8 🗣 💁 🕶 🖉 🔻 🖄	+ ₩ + *>	$\leftarrow + \rightarrow +$	
		Quic	k Access 🛛 😰 🖣	
🔳 🛛 🐾 🗖 🗖	the Packs ⊠	- 8	🗄 Outli 🛛 🗖 🗖	
🕀 🖻 🙀 🗸		🍇 🛷 🗢	€ ⊂ ▽	
🖻 🗁 ABOV Semic 🔺	Name	Descript *	CMSIS	
🛛 🗁 Active-Semi	AnalogDevices		🚏 5.3.0	
🖻 🗁 Ambiq Micr	a 🐸 ARM		5.2.0	
Analog Devi	AMP	Software	5.1.1	
	a 🌐 CMSIS (installed)	CMSIS ((5.1.0	
AutoChips	🔮 5.4.0 (131MB)	Aligned	5.0.1	
	😍 5.3.0 (installed)	Updated		
	5.2.0 (107MB)	CMSIS-C	• 4.5.0 • 4.4.0	
	5.1.1 (107MB)	CMSIS-R *	• 4.4.0	
Innicon			4.3.0	
Depris Service			4.1.1	
🗅 🗁 MediaTek 🍸			410	
< <u> </u>			• III • •	
🖹 Problems 📃 Conso	le 🛛	B. 🔝 🖻	🛃 🗉 🕶 🔂 🕶 🗖	
GNU ARM Eclipse Packs	console			
			* *	

图 2-15 更新器件包列表

2.2 软件配置

GNU MCU Eclipse 软件安装包中提供已配置的模板工程,导入软件安装 包中 eclipse_proj\demo\gpio 模板工程来配置 Eclipse 软件。

如果 GNU MCU Eclipse 安装包路径、JDK 安装路径、J-LINK 安装路径和高云云源软件安装路径与如上所述一致,则不需要配置 Eclipse 即可使用 Eclipse 软件进行编译、链接、烧录和调试功能。

2.2.1 导入模板工程

如图 2-16 所示,选择菜单栏 File 中的 Open Projects from File System, 导入模板工程 eclipse_proj\demo\gpio。

•	clipse_work - C/C++ - Eclipse		
File	Edit Source Refactor Navigate	Search Project	t Run Window Help
	New	Alt+Shift+N ►	onfigurations 🗸 on:
	Open File		· c · ⊗ · ☆ · 0 · 8 · 4 ·
	Open Projects from File System	- 1	Quick Access 📑 📴
	Close	Ctrl+W	
	Close All	Ctrl+Shift+W	×
	Save	Ctrl+S	
	Save As		An outline is not
R	Save All	Ctrl+Shift+S	available.
	Revert		
	Move		
-9	Rename	F2	
8	Refresh	F5	
	Convert Line Delimiters To	•	
₽	Print	Ctrl+P	
	Switch Workspace	÷	
	Restart		Console 🔲 Properties 🛛 😨 🗖 🗖
2	Import		
	Export		Resource Path
	Properties	Alt+Enter	
	Exit		

图 2-16 导入模板工程

2.2.2 选择 Properties

导入模板工程后,选择模板工程 gpio,右键选择 Properties,如图 2-17 所示。

Project Project Gradient Office Project Gradient Office Project Projec	2	Delete Remove from Context Source Move	Delete Ctrl+Alt+Shift+Down ▶
	P × 9	Rename	F2
	2	Export	
⊳ () () > () ()		Build Project Clean Project Refresh Close Project Close Unrelated Projects	F5
		Make Targets Index Build Configurations)
	*	Show in Remote Systems view Profiling Tools Run As Debug As Profile As Restore from Local History Run C/C++ Code Analysis Team Compare With	
-		Configure	•
🛎 gpio		Properties	Alt+Enter

图 2-17 选择 Properties

2.2.3 配置环境变量

选择 C/C++ Build 选项卡中 Environment 选项,如图 2-18 所示,配置如下环境变量:

- C:\ProgramData\Oracle\Java\javapath;
- C:\Windows\system32;C:\Windows;
- C:\Windows\System32\Wbem;
- C:\Windows\System32\WindowsPowerShell\v1.0\;
- D:\eclipse_tools\tools\GNU Tools ARM Embedded\5.4 2016q2\bin;

D:\eclipse_tools\tools\GNU MCU Eclipse\Build Tools\2.11-20180428-1604\bin

ype filter text	Environm	ent		$\langle \neg \bullet \neg \rangle \bullet \neg$
 Resource Builders C/C++ Build Build Variables 	Configur	ation: Debug [Active]	▼ Mana	age Configurations
Logging	Environr	nent variables to set		Add
Settings	Vari	Value	Origin	
Tool Chain Editor	CWD	D:\eclipse_tools\eclipse_prj\demo\gpio\Debug	BUILD SYSTEM	Select
VI C/CLL Compiler	PATH	C:\ProgramData\Oracle\Java\javapath;C:\Wind	USER: CONFIG	Edit
C/C++ General	PWD	D:\eclipse_tools\eclipse_prj\demo\gpio\Debug	BUILD SYSTEM	Delete
Linux Toois Path Project References Run/Debug Settings ▷ Task Repository	⊚ Appe ⊚ Repla	nd variables to native environment ce native environment with specified one		Undefine
WikiText			Restore Def	faults Apply

图 2-18 配置环境变量

2.2.4 配置工具链

选择菜单栏 Window 中的 Preferences,选择选项卡 C/C++中的 Build 下的 Workspace Tools Paths,配置工具链路径,如图 2-19 所示。

图 2-19 配置工具链



2.2.5 配置烧录工具

GNU MCU Eclipse 软件使用高云 Programmer 软件烧录 MCU 镜像文件。

在 GNU MCU Eclipse 软件中配置高云 Programmer 软件为 MCU 镜像文件烧录工具,选择菜单栏 Run 中 External Tools 下的 External Tools Configurations...,配置烧录工具,如图 2-20 所示。

图 2-20 配置烧录工具

Create, manage, and run configurations	
Run a program	Name: Download Main
	Arguments: device GW1NS-2Crun 22 -f F:\ec_workspace\ucos_iii\Debug\ucos_iii.bin

2.2.6 配置 JTAG 模式切换工具

如果用户需要调试项目工程,需要将JTAG模式由烧录模式切换为调试模式。

在 GNU MCU Eclipse 软件中配置高云 Programmer 软件为 JTAG 模式切换工具,选择菜单栏 Run 中 External Tools 下的 External Tools Configurations..., 配置切换工具, 如图 2-21 所示。

图 2-21 配置 JTAG 模式切换工具

Create, manage, and run configurations

😣 External tool location does not exist

	Name: Switch
type filter text	🖭 Main 🔗 Refresh 🗟 Build 🖾 Environment 🔲 Common
Q Program Q Download Q Switch	Location: D:\Gowin\1.8\programmer2\bin\programmer_cli.exe
	Working Directory:
	Arguments: device GW1NS-2Crun 23

图 2-22 交叉编译配置

ype filter text > Resource Builders • C/C++ Build Build Variables Environment Logging Settings Tool Settings • C/C++ Build Build Variables Environment Logging Settings Tool Chain Editor Tools Paths V C/C++ General Linux Tools Path Prefix: arm-none-eabi- Suffix: C compiler: gcc C++ compiler: g++ Archiver: ar Project References Hex/Bin converter: objcopy	r ARM_Test		x
 ▷ Resource Builders ▷ C/C++ Build Build Variables Environment □ Logging Settings □ Tool Chain Editor Tools Paths XL C/C++ Compiler ▷ C/C++ General Linux Tools Path Prefix: □ C/C++ General Linux Tools Path □ C/C++ General <li< th=""><th>Settings</th><th></th><th></th></li<>	Settings		
Nun/Jeoug settings ▷ Task Repository Listing generator: objdump WikiText Size command: size	d rriables nent ain Editor aths + Compiler eral Path Settings Tool Settin Name: Architecture: Prefix: Suffix: C compiler: C++ compiler Settings Hex/Bin conv Listing generi Size comman	Colchains Devices Build Steps Build Artifact Binary Parse Colors for ARM Embedded Processors (arm-none-eabi-gcc) ARM (AArch32) arm-none-eabi- gcc g++ ar er: objcopy r: objdump size	
Build command: make Remove command: rm Toolchain path: D:\Program Files\GNU Tools ARM Embedded5.4 2016q2\bin (to change it use the <u>global</u> or <u>workspace</u> preferences pages or the <u>project</u> propert Create flash image Create extended listing Print size	Build commar Remove com Toolchain pat Create flas Create extr V Print size	make nd: rm D:\Program Files\GNU Tools ARM Embedded5.4 2016q2\bin (to change it use the <u>global</u> or <u>workspace</u> preferences pages or the <u>project</u> propert mage led listing	

2.3 工程模板

2.3.1 工程创建

新建工程

选择菜单栏 File 中的 New 下的 C Project,如图 2-23 所示。

- 建立项目名称
- 选择项目类型 Empty Project
- 选择工具链 Cross ARM Gcc

图 2-23 新建工程

C Project Create C project of selected type	
Project name: limer	
rioject name. Jinter	
✓ Use default location	
Location: F:\ec_workspace\timer	Browse
Choose file system: default 👻	
Project type:	l oolchains:
GNU Autotools	Cross ARM GCC
Executable	Cross GCC
Empty Project	Microsoft Visual C++
Hello World UPC Project	XL C/C++ Tool Chain
Hello World ANSI C Project	
Hello World ARM C Project	
ADuCM36x C/C++ Project	
Hello World ARM Cortex-M C/C++ Project	
Freescale Kinetis KLxx C/C++ Project	
Freescale Processor Expert C/C++ Project	
SIM32F0xx C/C++ Project	
SIM32FI0x C/C++ Project	
SIM32F2xx C/C++ Project	
SIM32F3xx C/C++ Project	
SIM32F4xx C/C++ Project	
SIM32F/XX C/C++ Project	
Executable (XL OPC)	
Shared Library	
Static Library (AL OPC)	
Static Library (XL LIDC)	
Others	
Makefile project	
<	
Show project types and toolchains only if the	w are supported on the platform
end of the second and the second and the second sec	,
? sack	Next > Finish Cancel

选择配置

选择工程配置类型 Debug 和 Release,如图 2-24 所示。

图 2-24 选择工程类型配置

Select Configuration Select platforms and	ns d configurations you wish to deploy on	
Project type: Ex Toolchains: Cr Configurations:	ecutable oss ARM GCC	
☑ Š Debug☑ Š Release		Select all
		Deselect all
		Advanced settings
Use "Advanced sett	ings" button to edit project's properties.	
Additional configura Use "Manage config	ations can be added after project creation. gurations" buttons either on toolbar or on property	pages.
?	< Back Next > Fini	Sh Cancel

选择工具链和路径

选择交叉编译工具链 arm-none-eabi-gcc, 如图 2-25 所示。

图 2-25 选择工具链

Cross GNU ARM	Toolchain	
	ani and configure path	
Toolchain name:	GNU Tools for ARM Embedded Processors (arm-none-eabi-gcc)	•
Toolchain path:	D:\Program Files\GNU Tools ARM Embedded5.4 2016q2\bin	Browse
?	< Back Next > Finish	Cancel

建立工程结构

完成新建工程后,选择 Eclipse 工作空间下新建的项目工程,添加工程结构和代码,如图 2-26 所示。

- CORE: ARM Cortex-M3 内核定义
- PERIPHER: 外设驱动库
- STARTUP: 引导启动文件
- SYSTEM: 寄存器定义、系统初始化和系统时钟定义
- USER: 用户设计

• gw1ns2c_flash.ld: Flash 链接脚本

图 2-26 工程结构

\mu .settings
CORE
퉬 Debug
PERIPHER
STARTUP
SYSTEM
🐌 USER
cproject
project
gw1ns2c_flash.ld

导入文件

Eclipse 软件中,选择当前工程,右键选择 Refresh,自动导入如图 2-24 所示的工程结构和代码。

2.3.2 工程创建

Eclipse 软件中,选择当前工程,右键选择 Properties,选择选项卡 C/C++ Build,选择 Setting 选项,如图 2-27 所示。

filter text	Settings		← → →
lesource			
Builders			
C/C++ Build	Configuration: Debug [Active]		 Manage Configuration
Build Variables			
Environment	M Tool Settings M Toolshains	Ruild Steps	Ruild Artifact
Loading	Verteen settings Verteen anns Devices	P Duild Steps 2	Dulid Artifact
Settings	🚵 Target Processor	ARM family	cortex-m3
Tools Paths	🖄 Optimization	Architecture	Toolchain default
XL C/C++ Compiler	🖉 Warnings	Architecture	
C/C++ General	Debugging	Instruction set	Thumb (-mthumb)
inux Tools Path	Cross ARM GNU Assembler	Thumb interwor	k (-mthumb-interwork)
Project References	Preprocessor	Endianness	Toolchain default
Run/Debug Settings	Warnings		T 11 1 1 6 h
ask Repository	Miscellaneous	HOAT ABI	I ooichain default
VikiText	A S Cross ARM C Compiler	FPU Type	Toolchain default
	Preprocessor	Unaligned access	Toolchain default
	Includes		
	Dptimization	AArch64 family	Generic (-mcpu=generic)
	🖄 Warnings	Feature crc	Toolchain default
	🖄 Miscellaneous	Feature comto	Toolchain default
	Cross ARM C Linker	reatore crypto	
	General	Feature fp	Toolchain default
		Feature simd	Enabled (+simd)
	Wiscenareous		
	General	Lode model	Small (-mcmodel=small)
	Cross ARM GNU Print Size	Strict align (-mst	rict-align)
	A General	Other target flags	
			Restore Defaults Appl
			Appij

图 2-27 选择 Properties

配置 Cross ARM GNU Assembler Preprocessor

选择 Cross ARM GNU Assembler 中的 Preprocessor,配置汇编宏定义 ___STARTUP_CLEAR_BSS,如图 2-28 所示。



图 2-28 配置 Cross ARM GNU Assembler Preprocessor

配置 Cross ARM GNU Assembler Includes

选择 Cross ARM GNU Assembler 中的 Includes, 配置汇编引用文件路径, 如图 2-29 所示。



图 2-29 配置 Cross ARM GNU Assembler Includes

配置 Cross ARM C Compiler Preprocessor

选择 Cross ARM C Compiler 中的 Preprocessor, 配置 C 宏定义 USE_STDPERIPH_DRIVER, 如图 2-30 所示。



图 2-30 配置 Cross ARM C Compiler Preprocessor

配置 Cross ARM C Compiler Includes

选择 Cross ARM C Compiler 中的 Includes, 配置 C 引用文件路径, 如图 2-31 所示。



图 2-31 配置 Cross ARM C Compiler Includes

配置 Cross ARM C Linker

选择 Cross ARM C Linker 中的 General, 配置 Flash 链接脚本, 如图 2-32 所示。



图 2-32 配置 Cross ARM C Linker

配置 Cross ARM GNU Create Flash Image

选择 Cross ARM GNU Create Flash Image 中的 General, 配置 MCU 镜像 文件格式为 RAW binary, 如图 2-33 所示。



图 2-33 配置 Cross ARM GNU Create Flash Image

配置 Devices

选择 Devices 选项卡,选择器件 ARM Cortex-M3 "CMSDK_CM3",如图 2-34 所示。

Toor setur		is bettees	Juliu Steps		ong binary Par
evice sele	ction (Not yet used	d during build!)			
Name		Details			
⊿ ARI	м	Vendor			
Þ	ARM Cortex M33	(MPS Family (40	096 kB RAM, 819	2 kB ROM)	
4	ARM Cortex M3	Family (40	096 kB RAM, 409	6 kB ROM)	
	DS_CM3	Device (C	ortex-M3, r2p1,	25 MHz)	=
	CMSDK_CM3	Device (C	Cortex-M3, r2p1,	25 MHz)	_
\triangleright	ARM Cortex M33	Family (2	56 kB RAM, 4608	8 kB ROM)	
\triangleright	ARM Cortex M0	Family (40	096 kB RAM, 409	96 kB ROM)	
	ARM Cortex M0 p	lus Family (40	096 kB RAM, 409	6 kB ROM)	-
					,
Memory ma MSDK_CM Section	ap (Warning: Not y 3 Start	yet used to gene Size	erate the linker s	cripts!)	
Memory ma MSDK_CM Section	ap (Warning: Not) 3 Start	yet used to gene	Startup	cripts!)	
Memory ma MSDK_CM Section IRAM1	ap (Warning: Not) 3 Start 0x20000000	Size 0x00400000	Startup	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1	ap (Warning: Not) 3 Start 0x20000000 0x00000000	vet used to gene Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma CMSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not y 3 Start 0x20000000 0x0000000	Vet used to gene Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma CMSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x20000000 0x0000000	Vet used to gene Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not y 3 Start 0x2000000 0x0000000	vet used to gene Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x2000000 0x0000000	Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not y 3 Start 0x2000000 0x0000000	Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x20000000 0x0000000	Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x20000000 0x0000000	Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x20000000 0x00000000	vet used to gene Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x2000000 0x0000000	Size 0x00400000 0x00400000	Startup 0 1	cripts!)	
Memory ma MSDK_CM Section IRAM1 IROM1 Edit	ap (Warning: Not) 3 Start 0x2000000 0x0000000	Size 0x00400000 0x00400000 0x00400000	Startup 0 1	cripts!)	

图 2-34 配置 Devices

2.3.3 工程编译

完成工程配置和编码后,编译工程,选择工具栏编译按钮,开始编译, 如图 **2-35** 所示。

图 2-35 编译工程

File Edit Source	Refactor	Navigate	Search	Project	Run	Window	Help
< Q =	Sterr	al Tools	~ 💁	Downloa	d		
				1 e			
Project Explore	r 🕮 🍒 Co	nnections	- E A				
📋 free_rtos							
a 😂 lcd							
> 🐰 Binaries							
⊳ 🔊 Includes							
CORE							
⊳ 📂 Debug							
> 📂 PERIPHE	R						
D 🔁 STARTU	р						
D 🔁 SYSTEM							
Distribution Distribution Distribution							
gw1ns2d	_flash.ld						

2.3.4 工程烧录

完成工程编译后,将生成的镜像文件烧录到芯片中,选择工具栏运行按钮,调用已配置的烧录工具,如图 **2-36** 所示。

图 2-36 工程烧录



2.3.5 工程调试

配置调试选项

选择菜单栏 Run 中的 Debug Configurations 下的 GDB SEGGER J-Link Debugging 选项卡,右键选择 New,建立工程的调试配置选项,如图 2-37 所示。

图 2-37 建立调试配置选项

Debug Configurations	
Create, manage, and run configurations	TO T
Image: Second Secon	 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 'Filter' button to configure filtering options. Edit or view an existing configuration by selecting it. Configure launch perspective settings from the 'Perspectives' preference page.
Launch Group Qt Qt Local Application Filter matched 14 of 14 items	w plicate ete
?	Debug Close

选择已建立的调试配置选项。

1. 选择 Main 选项卡, 配置镜像文件路径, 如图 2-38 所示。

图 2-38 配置 Image 路径

Debug Configurations	
Create, manage, and run configurations	
	Name: Icd Debug
type filter text	📄 Main 🛛 🏇 Debugger 🕨 Startup 🦆 Source 🔲 Common
C C/C++ Application C C/C++ Attach to Application	Project: Icd
C/C++ Postmorte Debugger C/C++ Remote Application	C/C++ Application: Debug\cd.elf
OSF PDA Application C GDB Hardware Debugging GDB OpenOCD Debugging GDB OpenOCD Debugging GDB PyOCD Debugging	Build (if required) before launching Build Configuration: Select Automatically
C GDB QEMU Debugging C GDB SEGGER J-Link Debugging C Icd Debug Launch Group C Gui a Linu i i ui	 Enable auto build Use workspace settings
Qt Qt Local Application	

- 2. 选择 Debugger 选项卡, 配置 Debugger 器件名称和调试接口为 JTAG, 如图 2-39 所示。
- Device Name: Cortex-M3
- Interface: JTAG
- Other options: -singlerun -timeout

📄 Main 🕸 Deb	ugger 🔰 🕨 Startup 💱 Source 🔲 Common			
J-Link GDB Serv	er Setup			
V Start the J-L	nk GDB server locally			
Executable:	\${jlink_path}/\${jlink_gdbserver}			
Device name:	Cortex-M3			
Endianness:				
Connection:		ess)		
Interface:	🔘 SWD 💿 JTAG			
Initial speed:	O Auto ○ Adaptive Fixed 1000 kHz			
GDB port:	2331			
SWO port:	2332			
Telnet port:	2333			
Log file:				
Other options:	-singlerun -timeout			
Allocate con	sole for the GDB server			
GDB Client Setup				
Executable:	{cross_prefix}gdb{{cross_suffix}			
Other options:				
Commands:	Commands: set mem inaccessible-by-default off			

图 2-39 配置 Debugger

JTAG 模式切换

完成调试模式配置后,调用上述 Eclipse 软件配置中已配置的 JTAG 模式切换工具 Switch,完成 JTAG 软件模式由烧录模式切换到调试模式。

JTAG 接口切换

完成 JTAG 软件模式切换后,如果使用高云开发板"DK-EVAL-GW1NS V1.1",需要将开发板上的 JTAG 接口 TMS、TCK、TDI、TDO 四个跳线帽 由(1,3)端切换到(3,4)端口,完成 JTAG 接口切换,如图 2-40 所示。



图 2-40 JTAG 接口切换

调试

连接 J-LINK 仿真器,选择工具栏 Debug 按钮,开始调试,如图 2-41 所示。

图 2-41 启动调试

Debug 🛙 🙀 🚺	🗢 🖽 🗇 Variables 🕄 🗣 Break	points 🖫 Peripherals	8 4 B C 2 2	
c] Icd Debug (GDB SEGGER J-Link Debugging)	Name	Type	Value	
Mi source d'ann an l				
naine 81			🗄 Outline II 😤 Visualizer 🕼 🖻 🔩 💘 🔹 🗮	
* Copyright (C) 2012-2018 Gould Semiconductor Technology Go.110.[] * Taclades * Taclades * pictra: h* * Taclades * ends: ** * Taclade * ends: ** * Taclade * ends: ** * Taclade * ends: ** * **		6	Si golina(kok Si golina(kok) Si golina(kok) Si golina Si	
onsole 21 @ Taks (E Problems O Executables lebug (GDB SEGGER)-Link Debugging) Jink (DBServerCLese "Ling target CPU				
reskpoint reached g address bob000015C H synes g address bob000015C (tota = 0x47004800) mig trenkpoint g address bob000015C, Size = 2 vietg trenkpoint g address bob000015C, Size = 2 2 bytes g address bob000015C (biz = 0x4000) tota bob00015C (biz = 0x4000015C) (biz = 0x4000015C) tota bob00015C (biz = 0x4000015C) (biz = 0x400015C) tota bob00015C (biz = 0x400015C) (biz = 0x400015C) (biz = 0x400015C) tota bob00015C (biz = 0x400015C) (biz = 0x400015C) tota bob00015C (biz = 0x400015C) (biz = 0x400015C) tota bob00015C (biz = 0x400015C) (biz = 0x400015C) (biz = 0x400015C) tota bob00015C (biz = 0x400015C) (biz = 0x400015C) (biz = 0x400015C) tota bob00015C (biz = 0x400015C) (biz = 0x4000015C) (biz = 0x400015C) (biz = 0x4000015C) (biz = 0x400015C) (biz = 0x4000015C) (biz = 0x400015C) (biz = 0x400015C) (biz = 0x400015C) (biz = 0x400015C) (biz = 0x4000015C) (biz = 0x40000015C) (biz = 0x4000015C) (biz = 0x4000015C) (biz = 0x400000015C) (biz = 0x40000015C) (biz = 0x40000015C) (biz = 0x400000000				

2.4 参考设计

高云提供基于 GNU MCU Eclipse 软件的参考设计:

Gowin GW1NS-2C_MCU_PACK\Gowin_GW1NS-2C_MCU_RefDesign\ MCU_RefDesign\GNU

