1 Gowin Cortex-M1 软核使用说明

Gowin 软件中提供有 Cortex-M1 软核给用户使用,对单片机熟悉的用户来 说,Cortex-M1 并不陌生,本文档主要通过实验的方式带领读者学会使用 Gowin 软件提供的 Cortex-M1 进行软硬件编程,本次实验实现的功能就是通过 Cortex-M1 软核控制 GPIO 输出高低电平,从而控制 LED0 灯的亮灭。

1.1 硬件设计

1.1.1 Gowin 软件中添加 Cortex-M1 软核

我们首先在 Gowin 软件中新建一个工程,建立工程和文件的方法读者可以 参考逻辑教程中的"Gowin 软件基本开发流程"一节的内容,这里将不再进行 说明,我们这里新建一个名为"M1SC_5A_LED_Twinkle"的工程,然后点击

" 〕"新建一个 Verilog 文件,也命名为"M1SC_5A_LED_Twinkle"。

然后点击"[&]"进入 IP Core Generator 界面搜索"m1"找到 Cortex-M1 软 核"Gowin EMPU M1",如下图 1-1 所示。



图 1-1 找到 Cortex-M1 软核

双击之后,进入配置界面,如下图 1-2 所示。

小梅哥 FPGA 团队



专注于培养您的 FPGA 独立开发能力 开发板 培训 项目研发三位一体



图 1-2 Gowin_EMPU_M1 IP 配置界面

从上图可以看出, Gowin EMPU M1包括三级结构:

第一级, Cortex-M1 内核及 ITCM (指令存储器)、DTCM (数据存储器);

第二级, AHB 总线及 GPIO、CAN、Ethernet、DDR3 Memory、PSRAM Memory、SPI-Flash Memory、AHB Master [1-6];

第三级, APB 总线及 UART0、UART1、Timer0、Timer1、Watch Dog、 RTC、TRNG、DualTimer、I2C Master、SPI Master、SD-Card、APB Master [1-16]。

1.1.2 Cortex-M1 软核配置

进入 Gowin_EMPU_M1 IP 配置界面之后,双击 Cortex-M1,即可对 Cortex-M1 内核系统进行配置,如下图 1-3 所示。

小梅哥 FPGA 团队

武汉芯路恒科技

专注于培养您的 FPGA 独立开发能力 开发板 培训 项目研发三位一体



图 1-3 进入 Cortex-M1 内核系统配置

下面对 Cortex-M1 内核系统中各个配置进行说明。

1.1.2.1通用配置

Cortex-M1 通用配置如下图 1-4 所示。

Common Debug Memory					
Number of interrupts					
O 1					
0 8					
○ 16					
③ 32 ③					
✓ OS Extension					
Small Multiplier					
🗌 Big Endian					
External Interrupts					

图 1-4 Cortex-M1 通用配置选项

上述各项配置说明:

- Number of interrupts: 中断数量配置,可以配置1个、8个、16个或32 个外部中断,默认为32个。
- OS Extension:操作系统扩展配置,如果选择,则 Cortex-M1 扩展支持操作系统,默认为支持操作系统扩展。
- Small Multiplier:乘法器模式配置,如果选择,则Cortex-M1支持Small 店铺: <u>https://xiaomeige.taobao.com</u> 技术博客: <u>http://www.cnblogs.com/xiaomeige/</u> 官方网站: <u>www.corecourse.cn</u> 技术群组:

乘法器,否则支持 Normal 乘法器,默认为 Normal 乘法器。

- Big Endian: 数据存储格式配置,如果选择,则 Cortex-M1 支持数据大端格式,否则支持数据小端格式。
- External Interrupts:外部中断配置,如果选择,则 Cortex-M1 支持扩展的4个外部中断输入,否则不支持扩展的外部中断输入,默认为不支持。

1.1.2.2 调试配置

点击 Debug 进入调试配置界面,如下图 1-5 所示。

Common	Debug	Memory			
Enable	Debug				
 Serial Wire JTAG and Serial Wire 					
Small I	Debug				

图 1-5 Cortex-M1 调试配置

对上述配置说明如下:

- Enable Debug: 是否使能调试端口功能,勾选则使能,不勾选不使能。
- Debug Port Select: 调试端口的选择,一共有三个选项可选: JTAG、 Serial Wire、JTAG and Serial Wire。Serial Wire 就是"串行线调试",是 由两条线组成 SWDIO 和 SWCLK; JTAG Debug Port,也就是我们常用 的 J-link 上面的调试端口(JTAG 模式下),本次实验我们使用 DAP Link 上的调试端口,也就是 Serial Wire 模式。
- Small Debug: 调试模式配置,如果选择 Small Debug,则 Cortex-M1 支持 Small 模式调试器,否则支持 Full 模式调试器,默认为 Full 模式。

Debug 调试配置界面最终配置如下图 1-6 所示。

小梅哥 FPGA 团队 武汉芯路恒科技

专注于培养您的 FPGA 独立开发能力 开发板 培训 项目研发三位一体

Common	Debug	Memory			
Inable Debug					
Debug F	ort Select				
⊖ JTAG					
Serial Wire					
◯ JTAG and Serial Wire					
Small Debug					

图 1-6 调试配置

1.1.2.3存储配置

点击"Memory"进入存储配置界面,如下图 1-7 所示。

Common	Debug	Memory		
ITCM ITCM ITCM Inte Ext	ry iry			
ITCM Si:	ze: 32 KB lize ITCM itialization P	▼ ath:		
DTCM DTCM Select Internal Data Memory C External Data Memory				
DTCM S	ize: 32 KB	•		

图 1-7 Cortex-M1 存储配置界面

对上述配置说明如下:

- ITCM Select: ITCM 配置选择, ITCM 是 Cortex 内核中指令传输总线, 支持使用内部指令存储器"Internal Instruction Memory"或者外部指令 存储器 "External Instruction Memory"。
- ITCM Size: 只有选择了"Internal Instruction Memory",该配置才能被 启用,可以选择为1KB、2KB、4KB、8KB、16KB、32KB、64KB、 128KB、256KB 或 512KB; 根据不同的芯片类型支持的最大 Size 都是 不一样的, GW5A-25 的 ITCM Size 最大选择为 64KB, 默认 32KB。

小梅哥 FPGA 团队 武汉花路恒科技 专注于培养定的 FPGA 独立开发能力 开发板 培训 项目研发三位一体

- ITCM Initialization: ITCM 初始化配置,只有在勾选了"Initialize ITCM"的情况下,该配置才能被启用,如果选择 Initialize ITCM,则支持 ITCM 初始化,可以在 ITCM Initialization Path 导入 ITCM 初始值文件路 径。如果选择使用片外 SPI-Flash Memory下载引导方式,ITCM 初始值 根据不同的 ITCM Size 导入不同的 bootload 文件路径。注意 ITCM Initialization Path 导入的文件路径中,不能有以数字命名或"\r"、"\n" 等转义字符的文件夹路径,本次实验这里勾选使用"Initialize ITCM", 具体的初始化文件的获取将会在后面的内容进行说明,
- DTCM Select: DTCM 配置,可以选择"Internal Data Memory"或者 "External Data Memory",默认"Internal Data Memory"。Internal Data Memory: 内部数据存储器,片内 Block RAM 硬件存储资源,起始地址 0x2000000。External Data Memory: 外部数据存储器,如 DDRx Memory等,起始地址 0x20100000。
- DTCM Size: 当选择了"Internal Data Memory",该选项才可配置,可
 以选择 1KB、2KB、4KB、8KB、16KB、32KB、64KB、128KB、 256
 KB 或 512KB;不同芯片型号,支持的最大选择不同,GW5A-25 最大
 支持为 64KB,默认 32KB。

ITCM 是 Cortex 内核中指令传输总线,DTCM 是 Cortex 内核同 flash 及 sram 之间传输指令和数据的通道,指令的取指和执行及数据的读写在性能及管理上 存在差异性,由于是高速缓存,所以这两块内存区域被当做特殊的用途。比如 某些对时间要求非常严格的代码,就可以被放到 ITCM 中执行。这可以有效地 提高运行速度。某些需要频繁存取的数据,也可以放到 DTCM 中以节省存取时 间。本次实验将代码放到 ITCM 中去执行,本次实验的 Cortex-M1 存储配置如下 图 1-8 所示。

ITCM Select ● Internal Instruction Memory ○ External Instruction Memory ITCM Size: 32 KB ▼ ✓ Initialize ITCM ITCM Initialization Path:	
Internal Instruction Memory External Instruction Memory ITCM Size: 32 KB Initialize ITCM ITCM Initialization Path:	
 ○ External Instruction Memory ITCM Size: 32 KB ▼ ☑ Initialize ITCM ITCM Initialization Path: 	
ITCM Size: 32 KB ▼ Initialize ITCM ITCM Initialization Path:	
Initialize ITCM	
ITCM Initialization Path:	
DTCM	
DTCM Select	
 Internal Data Memory 	
○ External Data Memory	

店铺: <u>https://xiaomeige.taobao.com</u> 技术博客: http://www.cnblogs.com/xiaomeige/ 官方网站: <u>www.corecourse.cn</u> 技术群组: 图 1-8 Cortex-M1 存储器配置示意图

1.1.3 添加 GPIO 外设

在 Gowin_EMPU_M1 IP 的配置界面可以看到,该 IP 支持多种外设,AHB 总线上: GPIO、CAN、Ethernet、DDR3 Memory、PSRAM Memory、SPI-Flash Memory、AHB Master [1-6]; APB 总线上: UART0、UART1、Timer0、Timer1、Watch Dog、RTC、DualTimer、TRNG、I2C Master、SPI Master、SD-Card、APB Master [1-16]。双击便可以对外设进行使能配置,本次实验以 GPIO 为例, 双击进入 GPIO 配置界面,如下图 1-9 所示。



图 1-9 进入 GPIO 外设配置界面

进入配置界面之后,我们只需要使能 GPIO 即可,如下图 1-10 所示。



图 1-10 使能 GPIO 外设

点击 OK 之后,我们便可以看到 GPIO 选项变成了绿色,这也就表示外设配 店铺:<u>https://xiaomeige.taobao.com</u> 技术博客: http://www.cnblogs.com/xiaomeige/ 置成功,如下图 1-11 所示,然后点击 OK,Gowin_EMPU_M1 IP 配置就完成了。



图 1-11 外设配置成功

1.1.4 完成顶层模块代码设计

我们将 Gowin_EMPU_M1 IP 添加至顶层模块 "M1SC_5A_LED_Twinkle" 中,并将对应端口引出,最终 M1SC_5A_LED_Twinkle 文件中代码如下所示。



1.1.5 引脚约束

本次实验 Cortex-M1 的工作时钟 HCLK 设置为 50M,由开发板上的晶振提 供,复位 hwRstn 由开发板上按键控制,由于我们 ACG525 开发板上没有专门设 置关于 Cortex-M1 的调试接口,我们可以使用开发板上 40 pin 扩展 GPIO 的 GPIO[0]和 GPIO[1]来实现调试下载,其中 GPIO[0]连接 DAP-Link 上的 SWDIO, GPIO[1]连接 DAP-Link 上的 SWCLK。本次实验我们需要通过 Cortex-M1 软核控 制 GPIO 输出高低电平,从而控制 LED0 灯的亮灭,所以我们只需要分配 Cortex-M1 GPIO 外设的 GPIO[0]分配到 LED0 上即可。综上所述,可以得到本次实验的 引脚分配表如下表 1-1 所示。

引脚名称	引脚编号
HCLK	T9
hwRstn	C15
JTAG_7	C17
JTAG_9	C18
GPIO[0]	D14

表 1-1 引脚分配表

引脚分配完成之后,对工程进行全编译,无错误无警告,我们便可以进行 软件编程了。

1.2 软件代码设计

高云支持使用 ARM Keil MDK 或者他们自己的 CMD 软件进行软件代码设计,本次实验我们将带领读者使用 ARM Keil MDK 进行软件设计。关于 MDK 软件的安装和工程的建立,学习过单片机都会,如果读者对 MDK 软件使用不熟 悉请自行查找资料学习相关内容,这里我们将不做详细介绍,本次实验建立的 MDK 工程名为 led。

Gowin 针对 Gowin_EMPU_M1 提供软件编程库,读者可以在我们提供的例程中获取,也可以在高云官方网站上下载,下载方式如下图 1-12 所示。





在 MDK 软件中将工程建立完成之后,将提供的库文件添加至工程中,然后 建立 main.c 文件,然后编写本次实验需要的代码,如下所示:

/* Includes
*/
<pre>#include "GOWIN_M1.h"</pre>
/* Declarations
*/
<pre>void GPI0Init(void);</pre>

店铺: <u>https://xiaomeige.taobao.com</u> 技术博客: <u>http://www.cnblogs.com/xiaomeige/</u> 官方网站: <u>www.corecourse.cn</u> 技术群组: 专注于培养您的 FPGA 独立开发能力 开发板 培训 项目研发三位一体

```
void delay(__IO uint32_t nCount);
/* Definitions -----
----*/
int main(void)
{
 SystemInit(); //Initializes system clock
 GPIOInit(); //Initializes GPI00
 while(1)
 {
   GPI0_WriteBits(GPI00,0x0);
   delay(433300);
   GPI0_WriteBits(GPI00,0x1);
   delay(433300);
 }
}
//Initialize GPIO
void GPIOInit(void)
{
 GPIO InitTypeDef GPIO InitType;
 GPIO_InitType.GPIO_Pin = GPIO_Pin_0 |
                          GPIO Pin 1 |
                          GPIO_Pin_2
                          GPIO_Pin_3;
 GPI0_InitType.GPI0_Mode = GPI0_Mode_OUT;
 GPI0_InitType.GPI0_Int = GPI0_Int_Disable;
 GPI0_Init(GPI00,&GPI0_InitType);
 GPI0_WriteBits(GPI00,0xF);//light : low level
}
//delay
void delay(__IO uint32_t nCount)
{
 for(; nCount != 0; nCount--);
}
```

上述代码也非常好理解,首先初始化时钟,然后初始化 GPIO 口,然后在 while 循环中通过 GPIO WriteBits 函数控制对应 GPIO 口输出高低电平,使用的 是 GPIO0,并通过延时实现 LED 灯闪烁的功能。

1.2.1 MDK 软件配置

我们需要在 MDK 中进行相关设置才能启动 Cortex-M1 中程序的运行。我们 在前面讲解 Gowin_EMPU_M1 IP 配置的时候,我们提到使用 ITCM 的方式,将 代码放到 ITCM 中去执行,我们需要对 MDK 进行设置相关命令参数使其生成需 要的 itcm 文件。点击^这,对 MDK 工程进行相关配置。

1. 设置 ROM 的初始地址为 0x0

找到 Target 选项,将片上 IROM1 的起始地址设置为 0x0,如下图 1-13 所示。 如果这里不修改为 0x0, Coretx-M1 软核无法启动。

V Options for Target 'led'						
Device Target Output Listing User C/C++ Asm Linker Debug Utilities						
ARM ARMCM1 ARM Compiler: Use default compiler version 5 💌						
Operating system: None Use Cross-Module Optimization						
System Viewer File:		🗌 Us	e MicroLII	в 🗆 В	ig Endian	
Use Custom File						
Read/Only Memory Areas	'	Read/\	Write Merr	iory Areas		
default off-chip Start Size	Startup	default	off-chip	Start	Size	Nolnit
ROM1:	0		RAM1:			
□ ROM2: □	0		RAM2:			
ROM3:	0		RAM3:			
on-chip			on-chip			
IROM1: 0x0 0x7C00	•	◄	IRAM1:	0x20000000	0×8000	
IROM2:	0		IRAM2:			
OK	Canc	el	Def	aults		Help

图 1-13 设置 ROM 初始地址

2. 输出文件设置

找到 User 选项,在 After Build/Rebulid 选项下进行设置,添加如下命令: H:\keil\ARM\ARMCC\bin\fromelf.exe --bin -o led.bin .\Objects\led.axf ITCM\make_hex.exe led.bin

需要注意的是,上述命令中的"H:\keil\ARM\ARMCC\bin\fromelf.exe"需要根据读者电脑上 Keil 的安装路径进行修改,make_hex.exe 软件是用来生成 ITCM 文件的,这个软件读者可以在我们提供的例程中获取,如下所示。

小梅哥 FPGA 团队

武汉芯路恒科技

专注于培养您的 FPGA 独立开发能力 开发板 培训 项目研发三位一体



图 1-14 复制 make_hex.exe 软件至自己的工程

1.2.2 编译工程

经过上述操作之后,点击 〇,编译整个工程,直至整个工程无错误,如下

图 1-15 所示。



图 1-15 编译工程

通过上述操作,对于 Gowin Cortex-M1 软核的软硬件编程就完成了,接下来就是程序的下载和验证。

1.3 程序下载及功能验证

本章将带领读者使用两种方式下载,第一种就是 Gowin 软件和 MDK 软件 分开下载,这种方式适合于程序开发环节,需要调试的时候使用;第二种就是 将 MDK 软件中生成的 itcm 文件放至 Cortex-M1 软核中,然后编译工程,将生成 的 fs 文件下载至 FPGA 芯片中,这种方式适合程序开发完成,代码已经测试没

有问题,需要量产的时候使用。

1.3.1 MDK 与 Gowin 软件分开下载

在使用 MDK 软件下载程序的时候,需要使用到 DAP-Link 调试器或者其它 调试器,本次实验使用 DAP-Link,将 DAP-Link 的 GND、SCK、SWD 三个引脚 连接至开发板上,前面我们提到过,开发板上没有专用的调试接口,使用的是 40-pin 拓展接口的 GPIO[0]、GPIO[1],所以我们用三根杜邦线将 DAP-Link 和 ACG525 开发板进行连接,然后还需要使用高云下载器下载 FPGA 程序,所以本 次分开下载的硬件连接图如下图 1-16 所示。



图 1-16 分开下载硬件连接图

1.3.1.1 下载 FPGA 侧代码

需要注意的是,分开下载程序的时候,必须先下载 FPGA 侧的代码,然后下载 ARM 侧的代码,顺序如果反了,在 MDK 软件中将会检测不到调试器。我们这里打开 FPGA 侧的工程点击^{操 Programmer}下载程序,如下图 1-17 所示。

小梅哥 FPGA 团队 武汉花路恒科技 专注于培养定的 FPGA 独立开发能力 开发板 培训 项目研发三位一体

esign summary			Durain at Film		111/01	-1/11/00 54		-
🎼 User Constraints			Project File:		H:\01_gaoyun\02_n	MISC_SA_	LED_1 部选择	
FloorPlanner	1	🎎 Gowii	Programmer Version 1.	9.9 Beta-6 bu	ild 31030		- 0	×
🔀 Timing Constraints Editor	F	ile Edi	t Tools About	2				
📀 Synthesize		Q 4	3-8 1 I 🔗	事 USB	Cable Setting			
📗 Synthesis Report	Γ	Enable	Series	Device	Operat	ion	FS File	
📗 Netlist File	1		GW5A	GW5A-25A	SRAM Program		le/M1SC_5A_LED_Twinkle/fpga_	project/i
🥝 Place & Route								
📗 Place & Route Report								
📗 Timing Analysis Report								
Ports & Pins Report								
Programmer 1	<							>
	0	utput						đΧ
		nfo	Target Cable: Gowin USE	3 Cable(FT2CI	H)/0/337/null@2.5MHz			^
		nfo	Target Device: GW5A-25	A(0x0001281	3)			
uine o Uinersky	-1.	nfo	Operation "SRAM Progra	am" for devic	e#1			
esign Process Hierarchy	11	nfo	Frequency Updated: "15	MHz"				
nsole	1	nfo	User Code: 0x0000D2EB			3.	下载成功提示信息	- 11
	٦,	nfo	Status Code: 0x70026020)				
		nfo	Cost 3.54 second(s)					
								~
	R	eady						

图 1-17 下载 FPGA 侧代码

1.3.1.2下载 ARM 端代码

在 MDK 软件中,首先点击 ^杀进入 Debug 选项,找到 Use 选项,选择自己 使用的调试器,我们使用的 DAP-Link,这里就需要选择为 "CMSIS-DAP Debugger",如下图 1-18 所示。

<u>n n n</u>	🎠 🛊 🛱 //# //# 🖄 asc2_1608 🔍	🗟 🌾 🔍 • 🛛 • · 🔗 🍕 • 🛛 🖻 •
İ 🕺 📥	n 🔁 🗇 🐡 🏟	
nain.c 🗋 sta		
* @file * @authe	🐰 Options for Target 'led'	×
* @devi	Device Target Output Listing Vser C/C++ .	Asm Linker Debug Vtilities
*******	C Use Simulator <u>with restrictions</u> Settings	
L */	Limit Speed to Real-Time	ULINK Pro Cortex Debugger
#include	✓ Load Application at Startup ✓ Run to main()	Load CMSIS-DAP Debugger main()
/* Decla	Initialization File:	Initializatid Models Cortex-M Debugger
void GPI(void del:	Edit	.\ext_det ST-Link Debugger Edit
/* Defin	Restore Debug Session Settings	Restore SiLabs UDA Debugger
int main	Breakpoints Toolbox	Bre Altera Blaster Cortex Debugger
System	Watch Windows & Performance Analyzer	Watch Windows
GFIOIN		
↓ while(CPU DLL: Parameter:	Driver DLL: Parameter:
GPIO delaj	SARMCM3.DLL	SARMCM3.DLL
GPIO	Dialog DLL: Parameter:	Dialog DLL: Parameter:
dela		
}	Warn if outdated Executable is loaded	Wam if outdated Executable is loaded
//Initia void GPI	Manage Component Vi	ewer Description Files
GPIO_I	OK	Defaults Help

图 1-18 选择自己使用的调试器

然后点击 Settings,进入设置界面,我们就可以看到软件检测到了调试器,如下图 1-19 所示,表示此时我们可以下载程序了。

 \times

CMSIS-DAP Cortex-M Target Driver Setup

Any 👻		IDCODE	Device Name		Move
Serial No: LU 2022 8888	SWDIO	⊙ 0x2BA01477	ARM CoreSight SW-	DP	Up
imware Version: 1.0					Down
SWI Port: SW -	C Aut	omatic Detection	ID CODE:		
	C Ma	nual Configuration	Device Name:		
Max Clock: 1MHz	Add	Delete U	pdate	AP: 0x0	0
Connect & Reset Options			Cache Options	Download Options	
Connect: Normal 🗨 Re	set: Autodete	ect 💌	Cache Code	🔲 Verify Code Do	wnload
Reset after Connect			Cache Memory	Download to F	ash
Log Debug Accesses	Stop after Re	eset			

图 1-19 软件检测到了调试器

然后点击 🔍 、进入调试界面, 然后点击 🗐, 让程序开始运行, 如下图

1-20 所示。

🔀 H:\01_gaoyun\02_m1\M1SC_5A_LED_Twinkle\M1SC_5A_LED_Twinkle\m1_project\led\PROJECT\led.uvprojx - µVision					
File Edit View	Project Flash Deb	ug Peripherals Tools SVCS Window Help			
i 🗋 💕 🛃 🥔	X 🖻 🕵 🤊 🤆	* ← → 🏞 龍 龍 龍 譯 揮 /// /// /// asc2_1608 🛛 😡 🗟 🌾 🔍 🔍 • 🔿 🚓 • 🗐			
👬 🗟 🚳	円 () べ) ♦				
Registers Z	д 🔀	Disassembly			
Register □ Core R1 R2 R3 R4 E5 R6 R6 R7 R8 R8 R9 R10 R11 R12 R13 SP R16 R5 Backed R5 □ R15 (PC) □ R15 (PC)	Value 0::2000408 0::2000404 0::20000404 0::0000000 0::00000038 0::00000038 0::00000038 0::00000000 0::00000000 0::00000000 0::00000377 0::0000038 0::00000398 0::00000000 0::00000000 0::00000000 0::00000000 0::00000000 0::00000000 0::00000000 0::000000000 0::000000000 0::000000000 0::000000000 0::000000000 0::0000000000 0::00000000000000 0::0000000000000000000000000000000000	0x00000346 BD08 POP {r3,pc} 20: SystemInit(); //Initializes system clock c)0x00000348 FFFFFFES BL.W 0x0000031C SystemInit 21: GPIOInit(); //Initializes GPI00 22: 0x0000034C F7FFFFF BL.W 0x0000020E GPIOInit * */ */ mmmx 1 statup_GOWIN_MI.s */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ * */ */ *			
J. Statk	na.	22 23 while(1)			

图 1-20 ARM 端程序下载

程序运行起来之后,可以看到开发板上的 LED0 开始闪烁,代表本次实验 成功。

1.3.2 Gowin 软件下载完整工程代码

前面我们提到过,我们可以将 ARM 端的代码放到 ITCM 中去执行,也就是 将 ARM 的代码作为初始化文件在 Cortex-M1 中进行配置,从而就可以实现脱离 调试,直接通过 Gowin 软件下载完整的工程代码,具体操作如下所示。

1. 通过前面我们在 MDK 中的配置,可以看到在 MDK 工程目录下,生成 官方网站: www.corecourse.cn 店铺: <u>https://xiaomeige.taobao.com</u> 技术群组: 技术博客: http://www.cnblogs.com/xiaomeige/

了4个ITCM文件,将这4个文件进行复制。



图 1-21 复制 ITCM 文件

2. 在 FPGA 工程的 Cortex-M1 IP 目录下新建一个 itcm 文件夹,将刚刚复 制的 4个 itcm 文件放至该文件夹下,如下图 1-22 所示。



图 1-22 将 itcm 文件放置 FPGA 工程中

3. 在 Gowin 软件的 IP Core Generator 中点击 , 找到之前建立的 Cortex-M1 软核,如下图 1-23 所示。

🗠 👀 🔶ŭ 💷 🖽 🗠	
Target Device: GW5A-LV25UG	324C2/11 5
Filter	
Jame	🗱 Select IP Config file 🛛 🕹 🗙
Hard Module	← → ヾ ↑ □ « src gowin_e > ∨ ひ 搜索"gowin_empu_m1"
> ADC > BandGap	组织 ▼ 新建文件夹
> 🔚 CLOCK > 📒 DSP	♪ ^{音乐} ^ □ ^{itcm} 2. 找到建立的Cortex-M1软核
> I3C > IO > MIPI_DPHY > Memory > SPMI > User Flash > Soft IP Core > Al	■ 異回 *

图 1-23 找到建立的 Cortex-M1 软核 IP

4. 点击 gowin_empu_m1.ipc 文件,进入 Cortex-M1 配置界面,在 Memory 配置界面下,找到 ITCM 初始化文件配置,点击 找到我们刚刚建立 的存放 itcm 文件的文件夹路径,如下图 1-24 所示。



图 1-24 初始化 ITCM 文件

通过上述操作之后,点击 OK 重新生成 IP,然后重新进行分析和综合,生成新的 fs 文件。

然后将板子硬件连接好,硬件连接如下图 1-25 所示,按照 1.3.1.1 节的内容 重新下载新生成的 fs 文件。



图 1-25 硬件连接图

下载完成之后,可以看到开发板上的 LED0 开始闪烁,这也就说明我们本次实验通过配置 Cortex-M1 的 ITCM 文件下载程序成功。

1.4 思考与总结

我们通过一个简单的 LED 灯闪烁的实验,带领读者学会了使用 Gowin 提供的 Cortex-M1 软核进行软硬件代码设计,并在最后下载程序的时候,教给读者两种程序下载的方法,两种方法使用情况不一样,读者根据实际情况,选择适合的方式进行下载。读者学习完本文档的基本流程之后,自己可以通过 Cortex-M1 软核去设计更多丰富且有意义的工程。