

# Gowin\_EMPU(GW1NS-4C) IDE Software

# **Reference Manual**

IPUG928-2.0E, 03/14/2024

## Copyright © 2024 Guangdong Gowin Semiconductor Corporation. All Rights Reserved.

**GOWIN** is a trademark of Guangdong Gowin Semiconductor Corporation and is registered in China, the U.S. Patent and Trademark Office, and other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders. No part of this document may be reproduced or transmitted in any form or by any denotes, electronic, mechanical, photocopying, recording or otherwise, without the prior written consent of GOWINSEMI.

## Disclaimer

GOWINSEMI assumes no liability and provides no warranty (either expressed or implied) and is not responsible for any damage incurred to your hardware, software, data, or property resulting from usage of the materials or intellectual property except as outlined in the GOWINSEMI Terms and Conditions of Sale. GOWINSEMI may make changes to this document at any time without prior notice. Anyone relying on this documentation should contact GOWINSEMI for the current documentation and errata.

## **Revision History**

Date	Version	Description
04/20/2020	1.0E	Initial version published.
		AHB PSRAM Memory Interface peripheral supported.
	1.1E	AHB HyperRAM Memory Interface peripheral supported.
02/08/2021		APB SPI Nor Flash peripheral supported.
02/08/2021		GPIO supports multiple port types.
		<ul> <li>I<sup>2</sup>C supports multiple port types.</li> </ul>
		• ARM Keil MDK and GOWIN MCU Designer versions upgraded.
12/16/2022	1.2E	Software development kit updated.
03/14/2024	2.0E	Software programming reference design updated.

## Contents

Сс	ontentsi
Li	st of Figuresii
1	ARM Keil MDK1
	1.1 Software Installation1
	1.2 Project Template
	1.2.1 Create a New Project1
	1.2.2 Configuration Option
	1.2.3 Build
	1.2.4 Download
	1.2.5 On-line Debug
	1.3 Reference Design
2	GOWIN MCU Designer 15
	2.1 Software Installation
	2.2 Project Template
	2.2.1 Create a New Project
	2.2.2 Configuration Option
	2.2.3 Build
	2.2.4 Download
	2.2.5 On-line Debug
	2.3 Reference Design

## **List of Figures**

Figure 1-1 Create a New Project	2
Figure 1-2 Device Configuration	2
Figure 1-3 ROM and RAM Configuration	3
Figure 1-4 Output File Format Configuration	4
Figure 1-5 Header File Path Configuration	5
Figure 1-6 Flash Configuration	6
Figure 1-7 Build	7
Figure 1-8 Configuration Options for GW1NS-4C or GW1NSR-4C	8
Figure 1-9 Configuration Options for GW1NSER-4C	9
Figure 1-10 MCU JTAG Mode Configuration for GW1NS-4C or GW1NSR-4C	10
Figure 1-11 MCU JTAG Mode Configuration for GW1NSER-4C	11
Figure 1-12 Emulator Type Configuration	12
Figure 1-13 Debug Interface Type Configuration	13
Figure 1-14 Start Software Debugging	14
Figure 2-1 Create a New Project	16
Figure 2-2 Select Platform and Configuration	16
Figure 2-3 Select Toolchain and Path	17
Figure 2-4 Target Processor Configuration Option	18
Figure 2-5 Cross ARM GNU Assembler > Preprocessor Configuration Option	19
Figure 2-6 Cross ARM C Compiler > Includes Configuration Option	20
Figure 2-7 Cross ARM C Linker Configuration Options	21
Figure 2-8 Cross ARM GNU Create Flash Image Configuration Option	22
Figure 2-9 Devices Configuration Option	23
Figure 2-10 Build	24

Figure 2-11 Programmer Option	25
Figure 2-12 Configuration Options for GW1NS-4C/GW1NSR-4C	26
Figure 2-13 Configuration Options for GW1NSER-4C	27
Figure 2-14 Software Debug Level Configuration	28
Figure 2-15 Software Debug Configuration Option	29
Figure 2-16 Main Configuration Option	29
Figure 2-17 Debugger Configuration Option	31
Figure 2-18 MCU JTAG Mode Configuration for GW1NS-4C/GW1NSR-4C	32
Figure 2-19 MCU JTAG Mode Configuration for GW1NSER-4C	32
Figure 2-20 Start Software Debug	33

# **1** ARM Keil MDK

## **1.1 Software Installation**

For the detailed information, please refer to <u>*Getting Started with MDK*</u> (V5.26 and above) provided by ARM Keil MDK website.

## **1.2 Project Template**

ARM Keil MDK can be used for Gowin\_EMPU(GW1NS-4C) software programming. The steps include project creation, configuration, coding, build, download and debug.

## 1.2.1 Create a New Project

Open ARM Keil MDK and select "Project > New uVision Project..." to create a new project, as shown in Figure 1-1.

8	-			- <b>J</b>										
🔣 µVision											-	-		×
File Edit View	Pro	ect Flash	Debug	Peripherals	Tools S	VCS Wi	ndow	Help						
🗋 💕 🗟 🧭		New µVisio	n Project	•	2 12	微量	1 - E	/!≞ <i>  </i> ≩	1	RTOS		$\sim$	B. #	@ -
🖉 🖾 🗃 🤪		New Multi-	Project Wo	orkspace	81	5 5 4		<u> </u>						
Project		Open Proje	ect											
_		Close Proje	d											
		Export		)										
		Manage		)										
		Select Devis	ce for Targ	ot										
		Remove Ite	m											
	S.	Ontions		Alt+F7										
	1917	options		Altern										
		Clean Targe	ets											
		Build Targe	t	F7										
		Rebuild all	target file	s										
	1	Batch Build	i											
		Batch Setu	p											
	۲	Translate		Ctrl+F7										
Build Output		Stop build												<b>д X</b>
	_				_									^
														~
<														>
Create a new µVisio	n pro	ect												

Figure 1-1 Create a New Project

## **1.2.2** Configuration Option

## **Device Configuration**

ARM Cortex-M3 is embedded in Gowin\_EMPU(GW1NS-4C), and the device is configured as "ARM Cortex M3 > ARMCM3", as shown in Figure 1-2.

**Figure 1-2 Device Configuration** 



## **ROM and RAM Configuration**

Configure the start address and size of Flash for instructions and SRAM for data, as shown in Figure 1-3.

**ROM Configuration** 

- Start Address: 0x00000000
- Size: 0x8000 (32KB)

**RAM Configuration** 

- Start Address: 0x2000000
- Size: Consistent with the SRAM Size configuration of the Gowin\_EMPU(GW1NS-4C) IP Core in the IP Core Generator tool of Gowin Software, which can be configured as 2KB, 4KB, 8KB, or 16KB.

## Figure 1-3 ROM and RAM Configuration

🕅 Options for Target 'led'	×
Device Target Output Listing User   C/C++   As	sm   Linker   Debug   Utilities
ARM ARMCM3	Code Generation ARM Compiler: Use default compiler version 5
Operating system: None  System Viewer File: Use Custom File	Use Cross-Module Optimization Use MicroLIB Big Endian
Read/Only Memory Areas	Read/Write Memory Areas
default off-chip Start Size Startup	default off-chip Start Size Nolnit
□ ROM1: □ 0	□ RAM1: □
□ R0M2: ○	RAM2:
ROM3:	RAM3:
on-chip	on-chip
IROM1: 0x0 0x8000 €	▼ IRAM1: 0x20000000 0x2000 □
IROM2: C	IRAM2:
, ,,	, , , , , , , , , , , , , , , , , , , ,
OK Cano	el Defaults Help

## **Output File Format Configuration**

Gowin\_EMPU (GW1NS-4C) software programming design requires the generation of a software programming binary file. Therefore, it is necessary to convert the axf format to the binary format.

In the user command line option, convert the format of output file from axf to bin using fromelf, the output file format conversion tool. The call method is as shown in Figure 1-4.

The format conversion command of fromelf is: fromelf.exe --bin -o bin-file axf-file. For example, C:\Keil\_v5\ARM\ARMCC\bin\fromelf.exe --bin -o led.bin .\Objects\led.axf.

The path of fromelf should be modified according to the user's local installation path of ARM Keil MDK.

**Figure 1-4 Output File Format Configuration** 

evice   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				
Run #1		2	Not Specified	
Run #2		2	Not Specified	$\Box$
Before Build/Rebuild				
Run #1		2	Not Specified	$\Box$
Run #2		2	Not Specified	
After Build/Rebuild				
🔽 Run #1	D:\Keil_v5\ARM\ARMCC\bin\fromelf.exebin	2	Not Specified	$\Box$
🗌 Run #2		2	Not Specified	
Run 'After-Build' Conditionally				
I <u>B</u> eep When Complete	I Start Debugging			

## Header File Path Configuration

Configure C code header file path and different header file C paths are called during building, as shown in Figure 1-5.

For example:

- "..\..\library\libraries\cmsis\cm3\core\_support\mdk"
- "..\..\library\libraries\cmsis\cm3\device\_support"
- "..\..\library\libraries\drivers\inc"
- "..\..\library\middlewares\delay"
- "..\..\library\middlewares\gpio"

## • "..\inc"

## Figure 1-5 Header File Path Configuration

🔣 Options for Target 'led'					×
Device   Target   Output   L	isting User C,	/C++ Asm Linker Debu	ıg   Vtili	ties	
Preprocessor Symbols					
Language / Code Generatio		Strict <u>A</u> NSI C Enum <u>C</u> ontainer always int	<u>W</u> amings:	All Warnings	
Optimize for Time     Split Load and Store Mul     One <u>E</u> LF Section per Ful	tiple F	2lain Char is Signed Read-Onl <u>v</u> Position Independent <u>R</u> ead-Write Position Independent		No Auto Include C99 Mode GNU extensions	es s
Include Paths <u>M</u> isc Controls	raries/cmsis/cm3/con	e_support\mdk;\\.ibrary\libra	ries \cmsis \c	m3\device_suppc	
Compiler control string	ortex-M3 -li -g -O0ap raries/cmsis/cm3/core	ocs=interwork -split_sections -l e_support/mdk -l			<b>^</b>
	OK	Cancel Defaul	ts	Hel	.p

## **Flash Configuration**

If online debugging is required, disable "Utilities > Update Target before Debugging" option as shown in Figure 1-6.

5	
🕅 Options for Target 'led'	×
Device   Target   Output   Listing   Vser   C/C++   Asm   Linker   Debug   Utilities	
Configure Flash Menu Command	
Use Debug Driver Settings 🔽 Update Target before Debugging	
Init File:	
C Use External Tool for Flash Programming	
Command:	
Arguments:	
Run Independent	
Configure Image File Processing (FCARM):	
Output File: Add Output File to Group:	
user	J
Image Files Root Folder:	
OK Cancel Defaults H	elp

#### Figure 1-6 Flash Configuration

## 1.2.3 Build

After encoding and configuration, click "Build" (🔛) or "Rebuild" (🔛)

on tool bar, or click "Project > Build Targets" or "Project > Rebuild All Target Files" on the menu bar to generate software programming Binary File, as shown in Figure 1-7.

W	-		×
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject Fl <u>a</u> sh <u>[</u>	lebug Pe <u>r</u> ipherals <u>T</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp		
📄 💕 🛃 🍠 👗 🗈 🕵 🔊	(*) ← →   作 株 株 株 課 準 准 版   2 main	) 🚓  🖻	- 4
🛛 🕸 🕮 🗳 🕶 🔛 🛛 🚧 Ied	🖂 🔊 📥 🗟 🗇 🍏		
Project 🗜 🗵	main.c		▼ ×
🖃 🔧 Project: led	1 🗆 /*		^
🗄 😓 led	2	******	***
E liser	3 * @file main.c		
a and a second a seco	4 * @author GowinSemiconductor		
gwins4c_it.c	5 * @device Gowin_EMPU(GW1NS-4C)		
main.c	6 * @brief Main program body.		
🕀 🛄 driver	7 *************************************	******	***
🖻 🗁 cmsis	8 */		
core cm3.c	9 -		
	10 /* Includes		-*/
system_gw1ns4c.c	ll #include "gpio.h"		
startup_gw1ns4c.s	12 #include "delay.n"		
🖻 🦢 middleware	13		
delay.c	14 15 /# Definitions		*/
apio.c			/
	17 int main (void)		
	19 SystemThit(): //Initializes system clock		
	20 gpio init(); //Initializes GPIO		
	21 delay init(); //Initializes delay functions		
	22		~
E Pr ⟨♂Bo   {} Fu   0,→Te	¢		>
Build Output			д 🛛
linking			^
Program Size: Code=988 RO-	data=224 RW-data=20 ZI-data=1636		
FromELF: creating hex file			
After Build - User command	<pre>#1: D:\Keil_v5\ARM\ARMCC\bin\fromelf.exebin -o led.bin .\Objects\led.axf</pre>		- 6
".\Objects\led.axf" - 0 Er	ror(s), 0 Warning(s).		
Build Time Elapsed: 00:00	:07		~
<			>
	J-LINK / J-TRACE Cortex		

## Figure 1-7 Build

## 1.2.4 Download

Download the programming Binary file using Gowin Programmer.

Open Programmer in Gowin Software or under the installation path.

Click "Edit > Configure Device" on the menu bar or "Configure Device" ( $\implies$ ) on the tool bar to open the "Device configuration".

If FPGA product GW1NS-4C or GW1NSR-4C is selected, configuration options are as shown in Figure 1-8..

- Select "MCU Mode" in "Access Mode" drop-down list.
- Select "Firmware Erase, Program" or "Firmware Erase, Program, Verify" in "Operation" drop-down list.

🍓 Device configura	ition	?	×
-Device Operation -			
Access Mode:	MCV Mode		$\sim$
Operation:	Firmware Erase, Program		$\sim$
Firmware Erase, 1	Program		
-Programming Optio	ns		
File name: R4C_QN	48P_V1.1/gowin_empu/cm3_demo/imp	l/pnr/cm3_demo.fs	
User Flash Ini	tialization		
-FW/MCU/Binary Inp	ut Options		
-FW/MCU/Binary Inp Firmware/Binary F	ut Options ile: /Desktop/cm3_demo/project/l	ed/mdk_v5/led.bin	

Figure 1-8 Configuration Options for GW1NS-4C or GW1NSR-4C

If FPGA product GW1NSER-4C is selected, configuration options are as shown in Figure 1-9.

- Select "SecureFPGA Mode" in "Access Mode" drop-down list;
- Select "Firmware Erase, Program Securely" in "Operation" drop-down list.

Device conligura	tion ? ×
Device Operation	
Access Mode:	SecureFPGA Mode $\sim$
Operation:	Firmware Erase, Program Securely $\sim$
-Programming Optio:	ns
-Programming Option File name: R4C_QN	ns 48P_V1.1/gowin_empu/cm3_demo/impl/pnr/cm3_demo.fs
Programming Option File name: R4C_QN	ns 48P_V1.1/gowin_empu/cm3_demo/impl/pnr/cm3_demo.fs tialization
- Programming Option File name: R4C_QN User Flash Ini - FW/MCU/Binary Inp	ns 48P_V1.1/gowin_empu/cm3_demo/impl/pnr/cm3_demo.fs tialization ut Options
- Programming Option File name: R4C_QN User Flash Init - FW/MCU/Binary Inp Firmware/Binary Fi	ns 48P_V1.1/gowin_empu/om3_demo/impl/pnr/om3_demo.fs tialization ut Options ile: /Desktop/om3_demo/project/led/mdk_v5/led.bin

Figure 1-9 Configuration Options for GW1NSER-4C

- Import software programming Binary file in "FW/MCU/Binary Input Options > Firmware/Binary File".
- Click "Save" to complete the configuration.

#### Note!

Import hardware design bitstream in "Programming Options > File name", please refer to IPUG932, Gowin\_EMPU(GW1NS-4C) Hardware Design Reference Manual.

After device configuration, click Program/Configure () on the Programmer toolbar to complete the download of the software programming Binary file and the hardware bitstream file.

## 1.2.5 On-line Debug

After completing the download of the software programming Binary file and the hardware bitstream file, if there are any design issues, you can use U-LINK and J-LINK to debug online.

The single-step debug flow includes:

- Switch MCU JTAG mode
- Switch MCU JTAG interface
- Configure software debug
- Connect debug emulator

• Start software debug

## MCU JTAG Mode Switch

Switch JTAG mode from the downloading to debug mode using Programmer.

Open Programmer in Gowin Software or under the installation path.

Click "Edit > Configure Device" on the menu bar or "Configure Device" ( ) on the tool bar to open the "Device configuration".

If FPGA product GW1NS-4C or GW1NSR-4C is selected, MCU JTAG mode configuration options are as shown in Figure 1-10.

- Select "MCU Mode" in "Access Mode" drop-down list.
- Select "Connect to JTAG of MCU" in "Operation" drop-down list.

Figure 1-10 MCU JTAG Mode Configuration for GW1NS-4C or GW1NSR-4C

🙀 Device configuration			?	×
Device Operation				
Access Mode:	MCV Mode			$\sim$
Operation:	Connect to JTAG of MCU			$\sim$
Connect to JTAG of N	CV.			
		Save	Cano	el

If FPGA product GW1NSER-4C is selected, MCU JTAG mode configuration options are as shown in Figure 1-11.

- Select "SecureFPGA Mode" in "Access Mode" drop-down list.
- Select "Connect to JTAG of MCU" in "Operation" drop-down list.

🎄 Device configuratio	n	?	×
Device Operation			
Access Mode:	SecureFPGA Mode		$\sim$
Operation:	Connect to JTAG of MCU		~
Connect to Jiko of	<i>1</i> 0.0.		

Figure 1-11 MCU JTAG Mode Configuration for GW1NSER-4C

• Click "Save" to complete the configuration.

After device configuration, click "Program/Configure" () in the Programmer tool bar to complete MCU JTAG mode switch.

## MCU JTAG Interface Switch

Taking DK-START-GW1NSR4C-QN48G V1.1 for an instance, switch SW3, SW4, SW5, SW6 (TCK, TDO, TDI, and TMS of JTAG interface) on the development board from "FT232" (Download) to "J-LINK" (debug).

## Note!

- The development board must be kept on power in MCU JTAG mode and interface switch.
- If the development board powers off in the process, it will restore to MCU JTAG download mode automatically after power on.

## Software Debug Configuration Option

In the ARM Keil MDK software, click "Options for Target..." (<sup>(K)</sup>) on the tool bar to configure "Debug" option.

- Emulator Type Configuration Click the Debug emulator type drop-down list to configure type, as shown in Figure 1-12.
  - U-LINK Emulator
     If the U-LINK emulator is used, select "ULNK2/ME Cortex Debugger".
  - J-LINK Emulator If the J-LINK emulator is used, select "J-LINK/J-TRACE Cortex".

5 <b>11</b> 5	
🔣 Options for Target 'led'	×
Device   Target   Output   Listing   User   C/C++	Asm Linker Debug Utilities
C Use Simulator with restrictions Settings	Use: J-LINK / J-TRACE Cortex     Settings
Limit Speed to Real-Time	ULINKplus Debugger
I load Application at Startup         I load Application at Startup	Load     J_LINK / J-TRACE Cortex     Models Cortex-M Debugger     main()
Initialization File:	Initializatid ST-Link Debugger
Edit	NULink Debugger
Restore Debug Session Settings	Restore Stellaris ICDI
	SiLabs UDA Debugger
	TI XDS Debugger
V Watch Windows & Performance Analyzer	V watch windows
Memory Display Vewer	Memory Display Vesstem 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
Warn if outdated Executable is loaded	Wam if outdated Executable is loaded
Manage Component V	/iewer Description Files
OK Ca	uncel Defaults Help

Figure	1-12	Emulator	Type	Configuratio	n
Inguit	1-14	Linuator	Lypu	Connguiation	, 11

 Debug Interface Type Configuration Click Settings and open Cortex JLink/JTrace Target Driver Setup if J-LINK is selected, or ULINK2/ME Cortex-M Target Driver Setup if U-LINK is selected.

For example, the J-LINK emulator is selected, as shown in Figure 1-13.

- JTAG Debug Interface Gowin\_EMPU(GW1NS-4C) supports JTAG debug interface and configure Port to "JTAG".
- SW Debug Interface Gowin\_EMPU(GW1NS-4C) does not support SW debug interface, so do not select this one.

tex JLinl	k/JTrace Targ	get Driver Setup				
bug T	race Flash	Download				
J-Link /	J-Trace Adap	pter	JTAG Dev	ice Chain		
SN:	429496729	5 🔻		IDCODE	Device Name	IR len Move
Device:	J-Li	ink ARM	TDO	⊙ 0x4BA00477	ARM CoreSight JTAG-D	V 4 Up
HW :	V8.00	dll : V6.94	TDI	<		Down
FW : Po	J-Link ARM	V8 compiled No Max	C Autor	natic Detection	ID CODE:	
J <sup>2</sup>		Auto Clk	Add	Delete Up	odate IR len:	
Conne Conne I <u>R</u> e	ct & Reset O ct: Normal eset after Cor	Pptions Reset: No nnect	ormal	Cach	e Options Downlo Cache Code Veri Cache Memory Dov	ad Options fy Code Download vnload to <u>F</u> lash
Interfac	e B C TCP/IP	, TCP/IP Network S IP-Addres	Settings 	Port (A	Autodetect	Misc JLink Info JLink Cmd
		127	0.0	. 1 : (	D Ping	JLink Cmd

## **Figure 1-13 Debug Interface Type Configuration**

Disable "Verify Code Download" and "Download to Flash" options in "Download Options".

If the debug options configuration is successful, after switching MCU JTAG mode and interface and connecting the emulator, the IDCODE, Device Name and other information of Gowin\_EMPU(GW1NS-4C) will be displayed in the JTAG Device Chain".

## Start Software Debug

Connect the U-LINK or J-LINK debug emulator after software debug configuration.

Click the Debug "<sup>(Q)</sup>" on the tool bar, or click "Debug > Start/Stop Debug Session" on the menu bar to start software debug.

You can perform operations of breakpoint setting, single-step debug, reset, run, etc. as shown in Figure 1-14.

		- 🗆 ×
<u>F</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject Fl <u>a</u> sh <u>D</u> eb	ug Pe <u>r</u> ipherals <u>I</u> ools <u>S</u> VCS <u>W</u> indow <u>H</u> elp	
📄 🖬 😹 🖉 🕹 🛍 👘 🕲	*  🖛 →  陀 隐 隐 限  譯 譯 /// /// // 🖄 main 🛛 📃 🗟 🌮   🔍 🔹 🔗 🎸	t 🖃 – 🔍
않 🕲 🕙 🖓 🖓 🖓 🔅	N Q. R = Q. R - R - R - R - R - X - X - X	
Registers 📮 🖬 Dis	sassembly	<b>P</b> 🗙
Register Velue	0x0000046A 4001 DCW 0x4001	^
	<pre>19: SystemInit(); //Initializes system clock</pre>	
R0 0x08F	0x0000046C F7FFF6E BL.W SystemInit (0x0000034C)	
R1 0x200	<pre>20: gpio_init(); //Initializes GPIO</pre>	
R2 0x200   ➡>	0x00000470 F7FFFFE2 BL.W gpio_init (0x00000438)	
R3 0x200	<pre>21: delay_init(); //Initializes delay functions</pre>	~
N4 Ux000 <		>
R5 0x200		
R7 0x000	main.c startup_gw1ns4c.s	▼ ×
	17 int main (void)	^
R9 OxDEA	18 🖂 (	
R10 0x000	19 SystemInit(); //Initializes system clock	
R11 0x000	20 gpio_init(); //Initializes GPIO	
R12 0x200	21 delay_init(); //Initializes delay functions	
R14 (LR) 0x000	22	
R15 (PC) 0x000	23 while(1)	
± xPSR 0x210	24 🛱 - {	
t Banked	<pre>25 GPIO_ResetBit(GPIO0,GPIO_Pin_0); //LED1 on</pre>	
In System	26 delay_sec(1); //1 sec	
Mode Thread	27	
Privilege Privi	<pre>28 GPIO_SetBit (GPIO0, GPIO_Pin_0); //LED1 off</pre>	
Stack MSP 🔻	29 delay_sec(1);	×
🖻 Project 🚟 Registers <		>
Command	4 🛛 Call Stack + Locals	<b>д </b> 🗙
	^ Name Location/Value Type	
Load "C:\\Users\\liukai\\Des	ktop\\cm3_demo\\project\\led\\m	
* JLink Info: Reset: Halt com	re after reset via DEMCR.VC_COR main 0x0000000 intto	
* JLink Info: Reset: Reset de	evice via AIRCR.SYSRESETREQ.	
<	>	
>		
ASSIGN BreakDisable BreakEnak	ole BreakKill BreakList BreakSet 🛛 🖓 Call Stack + Locals 🔲 Memory 1	
	J-LINK / J-TRACE Cortex	t1: 0.00017390 se:

Figure 1-14 Start Software Debugging

## **1.3 Reference Design**

Gowin\_EMPU (GW1NS-4C) supports the reference design in ARM Keil MDK (tested software version V5.26), and you can click <u>here</u> to get the following reference design:

...\ref\_design\MCU\_RefDesign\MDK\_RefDesign\cm3\_demo、 cm3\_freertos、cm3\_rtthread\_nano、cm3\_ucos\_iii

# $2_{\rm GOWIN\,MCU\,Designer}$

## 2.1 Software Installation

The installation package of GMD is available at website <u>http://www.gowinsemi.com.cn/prodshow.aspx</u>.

## Note!

For the details of GMD, please refer to SUG549, GOWIN MCU Designer User Guide.

## 2.2 Project Template

GMD can be used for Gowin\_EMPU(GW1NS-4C) software programming. The steps include project creation, configuration, coding, building, download and debug.

## 2.2.1 Create a New Project

**Create a New Project** 

Select "New" ( $\square$ ) on the tool bar, or click "File > New > C Project" on the menu bar, as shown in Figure 2-1.

- Create a project name and location;
- Select the "Empty Project" type;
- Select the "Cross ARM GCC" build toolchain.

🐝 C Project	_		×
C Project			$\diamond$
Create C project of selected type			
Project name: gowin_led			
☑ Use <u>d</u> efault location			
Location: D:\GMD_workspace\workspace_cm1\gowin_led		B <u>r</u> owse	
Choose file system: default 🗸			
Project type: Toolchains:			
Executable     ARM Cross GCC			
Empty Project     RISC-V Cross GC	C		
Show project types and toolchains only if they are supported	d on th	ne platfo	rm
? < <u>B</u> ack <u>N</u> ext > <u>F</u> inish		Cance	el

## Figure 2-1 Create a New Project

## Select Platform and Configuration

Select "Debug" and "Release" as the platform and configuration, as

shown in Figure 2-2.

**Figure 2-2 Select Platform and Configuration** 

🐳 C Project		- 🗆 X
Select Configurations Select platforms and configuratio	ns you wish to deploy on	
Project type: Executable Toolchains: ARM Cross GCC Configurations:		
☑ 🛞 Debug ☑ 🛞 Release		Select all
Use "Advanced settings" button to Additional configurations can be Use "Manage configurations" but	o edit project's properties added after project creat tons either on toolbar or	s. ion. on property pages.
? < <u>B</u> ack	<u>N</u> ext > <u>F</u> in	ish Cancel

## Select Toolchain and Path

Select "arm-none-eabi-gcc" as the cross toolchain and its path, the default Toolchain name and Toolchain path are recommended, as shown in Figure 2-3.

Figure 2-3 Select Toolchain and Path

关 C Project			×
GNU ARM Cross	Toolchain		>
Select the toolch	ain and configure path		1
Toolchain name:	GNU MCU Eclipse ARM Embedded GCC (arm-none-eabi-gcc)		~
Toolchain path:	D:\GMD\toolchain\ARM_toolchain\bin	Brows	e
?	< <u>B</u> ack <u>N</u> ext > <u>F</u> inish	Cancel	

## **Create a Project**

After completing the project creation, navigate to the Project Explorer view, select the newly created project, add project structure and code, and import the software programming design.

In the Project Explorer view, select the current project, right-click and choose "Refresh" option to automatically update the structure and code of the current project.

## 2.2.2 Configuration Option

In the Project Explorer view, select the current project, right click and select "Properties > C/C++ Build > Settings" to configure the project parameters.

## **Target Processor Configuration**

Configure "Target Processor > ARM family", and set this option as "cortex-m3", as shown in Figure 2-4.

🛞 Tool Settings 🛞 Toolchains 📕 Devices	🎤 Build Steps 🧧	P Build Artifact 📓 Binary I 🔹 🕨
🖄 Target Processor	ARM family	cortex-m3
🖄 Optimization	Architecture	Toolchain default
🖉 Warnings	Architecture	
🖄 Debugging	Instruction set	Thumb (-mthumb)
V 🛞 GNU ARM Cross Assembler	Thumb interworl	k (-mthumb-interwork)
Preprocessor	Endianness	Toolchain default
2 Includes		
Warnings	Float ABI	Toolchain default
Solution Section Compiler	FPU Type	Toolchain default
Preprocessor	Unaligned access	Toolchain default
Includes	AArch64 family	Generic (-mcpu=generic)
🖉 Warnings	Feature crc	Toolchain default
🖄 Miscellaneous		
🗸 🛞 GNU ARM Cross C Linker	Feature crypto	loolchain default
🖉 General	Feature fp	Toolchain default
🖉 Libraries	Feature simd	Enabled (+simd)
Miscellaneous	reactive sinner	
GNU ARM Cross Create Hash Image	Code model	Small (-mcmodel=small)
General	Strict align (-mst	rict-align)
General     General	Other target flags	
	<	>

Figure 2-4 Target Processor Configuration Option

## **GNU ARM Cross Assembler > Preprocessor Configuration**

Configure "GNU ARM Cross Assembler > Preprocessor > Defined symbols (-D)" as "\_\_\_STARTUP\_CLEAR\_BSS" as shown in Figure 2-5.

🛞 Tool Settings	🛞 Toolchains	Devices	🎤 Build Steps	🚇 Build Artifact	🗟 Binary I 🔹 🕨
<ul> <li>Tool Settings</li> <li>Target Pro</li> <li>Optimizati</li> <li>Optimization</li> <li>Optimization<!--</td--><td><ul> <li>Toolchains</li> <li>Toolchains</li> <li>Cross or</li> <li>Cross Assemble</li> <li>Cross Assemble</li> <li>Cross Comp</li> <li>Cross C Comp</li> <li>Cross C Comp</li> <li>Cross C Linker</li> <li>Ianeous</li> <li>Cross C Linker</li> <li>al</li> <li>Cross Create F</li> <li>al</li> <li>Cross Print Size</li> <li>al</li> </ul></td><td>ler iler lash Image</td><td>Build Steps Use preproce Do not searc Preprocess of Defined symbol STARTUP_CLE Undefined sympol</td><td>Puild Artifact essor h system directori only (-E) ols (-D) E AR_BSS</td><td>Binary ( )</td></li></ul>	<ul> <li>Toolchains</li> <li>Toolchains</li> <li>Cross or</li> <li>Cross Assemble</li> <li>Cross Assemble</li> <li>Cross Comp</li> <li>Cross C Comp</li> <li>Cross C Comp</li> <li>Cross C Linker</li> <li>Ianeous</li> <li>Cross C Linker</li> <li>al</li> <li>Cross Create F</li> <li>al</li> <li>Cross Print Size</li> <li>al</li> </ul>	ler iler lash Image	Build Steps Use preproce Do not searc Preprocess of Defined symbol STARTUP_CLE Undefined sympol	Puild Artifact essor h system directori only (-E) ols (-D) E AR_BSS	Binary ( )
<ul> <li>♥ GNU ARM</li> <li> <sup>™</sup> S GNU ARM</li> <li>♥ S GNU ARM</li> <li> <sup>™</sup> S GNU ARM</li> <li></li></ul>	1 Cross Create F al 1 Cross Print Sizi	e			

Figure 2-5 Cross ARM GNU Assembler > Preprocessor Configuration Option

## **GNU ARM Cross C Compiler > Includes Configuration**

Select "GNU ARM Cross C Compiler > Includes > Include paths (-I)" to configure the C header file path, as shown in Figure 2-6.

Tool Settings 🛞 Toolchains 📕 Devices	🎤 Build Steps  🙅 Build Artifact  🗟 Binary Parsers 🛛 😣 Error Parsers
<ul> <li>Target Processor</li> <li>Optimization</li> <li>Warnings</li> <li>Debugging</li> <li>S GNU ARM Cross Assembler</li> <li>Preprocessor</li> <li>Includes</li> <li>Warnings</li> <li>Miscellaneous</li> <li>S GNU ARM Cross C Compiler</li> <li>Preprocessor</li> <li>Includes</li> <li>Optimization</li> <li>Warnings</li> <li>Miscellaneous</li> <li>S GNU ARM Cross C Linker</li> <li>General</li> <li>Libraries</li> <li>Miscellaneous</li> </ul>	Include paths (-I)       Image: Second Structure         *\$(workspace_loc:/\${ProjName}/src/library/libraries/cmsis/cm3/device_support)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/delay)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/delay)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/gpio)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/ppio)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/ppio)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/psram)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/psram)*         *\$(workspace_loc:/\${ProjName}/src/library/middlewares/spiflash)*         **(workspace_loc:/\${ProjName}/src/library/middlewares/spiflash)*         **
<ul> <li>♥ GNU ARM Cross Create Flash Image</li> <li>2 General</li> <li>♥ ♥ GNU ARM Cross Print Size</li> <li>2 General</li> </ul>	Include files (-include)
(꼳 General ☞ 꽝 GNU ARM Cross Print Size 꼳 General	Include files (-include)

## Figure 2-6 Cross ARM C Compiler > Includes Configuration Option

Take GMD\_RefDesign\cm3\_demo software programming reference design for an instance, the C header file paths are listed as follows.

- "\${workspace\_loc:/\${ProjName}/src/library/libraries/cmsis/cm3/device\_ support}"
- "\${workspace\_loc:/\${ProjName}/src/library/libraries/cmsis/cm3/core\_su pport/gmd}"
- "\${workspace\_loc:/\${ProjName}/src/library/libraries/drivers/inc}"
- \${workspace\_loc:/\${ProjName}/src/library/middlewares/delay}"
- "\${workspace\_loc:/\${ProjName}/src/library/middlewares/dmm}"
- "\${workspace\_loc:/\${ProjName}/src/library/middlewares/gpio}"
- \${workspace\_loc:/\${ProjName}/src/library/middlewares/hyper\_ram}"
- "\${workspace\_loc:/\${ProjName}/src/library/middlewares/psram}"
- "\${workspace\_loc:/\${ProjName}/src/library/middlewares/spi\_flash}"
- \${workspace\_loc:/\${ProjName}/src/library/middlewares/uart}"
- \${workspace\_loc:/\${ProjName}/src/project}"

## **GNU ARM Cross C Linker Configuration**

Configure the "GNU ARM Cross C Linker > General > Script files (-T)" option, and "gw1ns4c\_flash.ld" is used as the GMD Flash linker, as shown in Figure 2-7.





Take GMD\_RefDesign\cm3\_demo software programming reference design for an instance, the Flash linker configuration is as follows.

"\${workspace\_loc:/\${ProjName}/src/library/libraries/cmsis/cm3/device \_support/startup/gmd/linker/gw1ns4c\_flash.ld}"

## GNU ARM Cross Create Flash Image Configuration

Configure "Cross ARM GNU Create Flash Image > General > Output file format (-O)", and set this option as "RAW binary" to generate software programming Binary file, as shown in Figure 2-8.

🛞 Tool Settings 🛞 Toolchains 🔳 Devices	🎤 Build Steps 🚇 Build Artifact 🗟 Binary 🔹 🕨
<ul> <li>Tool Settings Toolchains Devices</li> <li>Target Processor</li> <li>Optimization</li> <li>Warnings</li> <li>Debugging</li> <li>S GNU ARM Cross Assembler</li> <li>Preprocessor</li> <li>Includes</li> <li>Warnings</li> <li>Miscellaneous</li> <li>S GNU ARM Cross C Compiler</li> <li>Preprocessor</li> <li>Includes</li> <li>Optimization</li> <li>Warnings</li> <li>Miscellaneous</li> <li>S GNU ARM Cross C Lompiler</li> <li>Miscellaneous</li> <li>S GNU ARM Cross C Linker</li> <li>General</li> <li>Libraries</li> <li>Miscellaneous</li> <li>S GNU ARM Cross Create Flash Image</li> <li>General</li> <li>S GNU ARM Cross Print Size</li> <li>General</li> </ul>	Build Steps Build Artifact Binary
	Other flags
< >	<

Figure 2-8 Cross ARM GNU Create Flash Image Configuration Option

## **Devices Configuration**

Configure "Devices > Device selection" option, and set this option as "ARM Cortex M3 > ARMCM3" in, as shown in Figure 2-9.

>	ARM Cortex A9	Family (204			
>	ARM Contour MO	) (	48 kB RAM, 204	8 kB ROM)	
>	ARIVI Cortex IVIO	Family (128	3 kB RAM, 256 l	(B ROM)	
r	ARM Cortex M0 plu	s Family (128	3 kB RAM, 256 l	(B ROM)	
>	ARM Cortex M1	Family (128	3 kB RAM, 256 l	(B ROM)	
>	ARM Cortex M23	Family (256	5 kB RAM, 4096	kB ROM)	
~	ARM Cortex M3	Family (128	3 kB RAM, 256 l	(B ROM)	
	ARMCM3	Device (Co	rtex-M3, Rev r2	p1, 10 MHz)	
>	ARM Cortex M33	Family (256	5 kB RAM, 4096	kB ROM)	
IRAM1	0x2000000	0x00020000	0		
IROM1	0x00000000	0x00040000	1		
Edit					

## 2.2.3 Build

After completing the project configuration and code writing, build the

current project; click "Build" ( $^{igsimes}$ ) or "Build All" ( $\overline{^{igsimes}}$ ) on the tool bar, or click

"Project > Build Project" or "Project > Build All" on the menu bar to build to generate the software programming Binary file, as shown in Figure 2-10.



#### Figure 2-10 Build

## 2.2.4 Download

Download software programming design BIN files using Programmer.

Click "Run> Programmer" on the GMD menu bar or Programmer "

workspace cm3 hard - C/C++ - cm3	demo/src/project/led	/led_demo.c - GOWIN MCU Designer	×
File Edit Source Refactor Navigate	Search Project	un Window Help	
	▶:⊕n:  ::	Programmer	
	D:"₩:♥#:1₽" *		8
Project Explorer 🛛 📃 🗖	led_demo.c ⊠	Run Programmer Ctrl+FII	
🖻 🔄 🗸	17	bebug FII	^
✓ 🖾 cm3_demo 🔨	18 /* Include 19 #include	Run History >	
> 🖑 Binaries	20 #include '	Run As	
> 🔊 Includes	21	Run Configurations	- 61
> 👝 Debug	23 /* Definit	Debug History >	
V 🔁 src	24 25 //Applicat	Debug As >	
> ibrary	26⊖ int led_de	Debug Configurations	
A abb2 master	27 { 28 SystemTr	Toggle Breakpoint Ctrl Shift P	
> 🔁 apb2 master	29 gpio_in:	Toggle breakpoint Cert-sint+b	
> 👝 dmm	30 delay_ir	Toggle Ene Breakpoint	
> 🗁 gpio_in_intr	32 while(1)	Toggle Wethbaint	
> 📂 hyper_ram	33 { 34 GPTO (	Skin All Breaknointr Ctrl+Alt+B	
> 👝 i2c_master		Remove All Breakpoints	× *
> 🔁 interrupt		Breakpoint Types	
> 👝 keyscan	🖹 Problems 🖉 🛄	preakpoint types prisole	- 8
> ied			• 📩 •
> 🕞 printi	CDT Build Console	:m3_demo]	
> 🗁 rtc	Tinished buildin	, carget. cms_demo.err	^
> 👝 spi_flash	Invoking: GNU AR	I Cross Create Flash Image	
> 🗁 spi_master	Finished buildin	copy -o binary cms_demo.ei; cms_demo.bin	
> 👝 timer	Tauahing, GNU AD		
> 🗁 uart_rx	arm-none-eabi-si	eformat=berkeley "cm3 demo.elf"	
> 🔁 uart_rx_intr	text data	bss dec hex filename	
> 🍃 wdog	Finished buildin	40 2044 ab4 cm5_demo.eit : cm3 demo.siz	
main c			
i cm3 freertos	<		>
		Writable Smart Insert 3:1	: 6
			: 🔳

#### Figure 2-11 Programmer Option

Click "Edit > Configure Device" on the menu bar or Configure Device "

If FPGA product GW1NS-4C/GW1NSR-4C is selected, configuration options are as shown in Figure 2-12.

- Select "MCU Mode" in "Access Mode" drop-down list.
- Select "Firmware Erase, Program" or "Firmware Erase, Program, Verify" in "Operation" drop-down list.

🏘 Device configura	ion	?	×
-Device Operation -			
Access Mode:	MCV Mode		$\sim$
Operation:	Firmware Erase, Program		$\sim$
-Programming Option	5		
File name: R4C_QN	8P_V1.1/gowin_empu/cm3_demo/impl/ ialization	/pnr/cm3_demo.fs	
-FW/MCU/Binary Inpu Firmware/Binary Fi	t Options le: //Desktop/cm3_demo/project/led	l/mdk_v5/led.bin	
		Save Ca	ncel

Figure 2-12 Configuration Options for GW1NS-4C/GW1NSR-4C

If FPGA product GW1NSER-4C is selected, configuration options are as shown in Figure 2-12.

- Select "SecureFPGA Mode" in "Access Mode" drop-down list.
- Select "Firmware Erase, Program Securely" in "Operation" drop-down list.

Device configura	ation	?	×
Device Operation			
Access Mode:	SecureFPGA Mode		$\sim$
Operation:	Firmware Erase, Program Securely		$\sim$
Programing Optio			
110gramming optic			
File name: R4C_Q1	N48P_V1.1/gowin_empu/cm3_demo/impl/pnr/cm3_d	lemo. fs	
File name: R4C_Q1 User Flash Ini FW/MCU/Binary Ing Firmware/Binary F	N48P_V1.1/gowin_empu/cm3_demo/impl/pnr/cm3_d tialization put Options file: /Desktop/cm3_demo/project/led/mdk_v5/1	lemo.fs	

#### Figure 2-13 Configuration Options for GW1NSER-4C

- Import software programming BIN file in "FW/MCU/Binary Input Options > Firmware/Binary File".
- Click "Save" to complete the configuration.

## Note!

Import hardware design bitstream in "Programming Options > File name", please refer to IPUG932, Gowin\_EMPU(GW1NS-4C) Hardware Design Reference Manual.

After device configuration, click Program/Configure (
) on the Programmer toolbar to complete the download of software programming Binary file and hardware bitstream file.

## 2.2.5 On-line Debug

After downloading software programming Binary file and hardware bitstream file, if there are any design issues, you can use J-LINK emulator to debug on-line ( The on-line debug design must be same with the one downloaded to the chip ).

The single-step debug flow includes:

- Software debug level configuration
- Configure software debug
- MCU JTAG mode switch

- MCU JTAG interface switch
- Connect debug emulator
- Start software debug

## Software Debug Level Configuration

In the Project Explorer view, select "Properties> C/C + + Build> Settings> Settings> Tool Settings> Debugging> Debug level", and the level Default(-g) or Maximum(-g3) is recommended, as shown in Figure 2-14.

Figure	2-14	Software	Dehuø	Level	Config	urstion
riguit	4-14	Soltware	DUDUg		Coning	ui ation

🛞 То	ol Settings	🛞 Toolchains	Devices	🎤 Build Steps	ዋ Buil	d Artifact	🗟 Binary	• •
Č	Target Pr	ocessor		Debug level		Default (-	g)	$\sim$
l 🖉	Optimizat	tion		Debug format		Toolchain	default	~
<u>é</u>	🤌 Warnings	;				·	uclaut	
(Å	🎐 Debuggir	ng		Generate pro	finform	ation (-p)		
× 8	🖇 GNU ARN	A Cross Assemb	ler	Generate gpr	of inform	mation (-p	g)	
	🖄 Prepro	ocessor		Other debuggin	g flags			
	🖄 Includ	es						
	🖄 Warni	ngs						
	🖄 Misce	llaneous						
× 8	GNU ARN	A Cross C Comp	iler					
	🖄 Prepro	ocessor						
	🖄 Includ	es						
	🖄 Optim	ization						
	🖄 Warni	ngs						
	🖄 Misce	llaneous						
× 8	GNU ARN	A Cross C Linker						
	😕 Gener	al						
	😕 Librari	ies						
		llaneous	1.1.1					
× 8		vi Cross Create F	lash image					
~ 8		a A Cross Drint Siz	_					
¥ 0		ol						
	Bener	ai						
				-				

## Software Debug Configuration Option

Select "Run > Debug Configurations > GDB SEGGER J-Link Debugging " and right-click "New" to create the project debug configuration options, as shown in Figure 2-15.

Figure 2.1	5 Softwar	e Dehuo	Configuration	Ontion
Figure 2-1	15 SULWAL	e Debug	Configuration	Option

🗱 Debug Configurations		×
Create, manage, and run configuration:	5	Ú.
Image: Second Secon	Configure launch settings from this dialog: <ul> <li>Press the 'New' button to create a configuration of the selected type.</li> <li>Press the 'Duplicate' button to copy the selected configuration.</li> <li>Press the 'Delete' button to remove the selected configuration.</li> <li>Press the 'Filter' button to configure filtering options.</li> <li>Edit or view an existing configuration by selecting it.</li> </ul> <li>Configure launch perspective settings from the 'Perspectives' preference page.</li>	
?	Debug	Close

Select the created debug options.

1. Select "Main" to configure the Project and C/C++ Application options of the current debugging project, as shown in Figure 2-16.

Figure 2-16 Main Configuration Option

° 🗎 🗶 🕒 🕆 ▼	Name: cm3 demo Debug	
ype filter text © C/C++ Application © C/C++ Attach to Application © C/C++ Postmortem Debugger © C/C++ Remote Application © GDB Hardware Debugging © GDB OpenOCD Debugging © GDB OpOCD Debugging © GDB OpOCD Debugging ♥ C GDB SEGGER J-Link Debugging ♥ c m3_demo Debug ▶ Launch Group	Main Startup Project: Cm3_demo C/C++ Application: Build (if required) before launching Build Configuration: Select Automati © Enable auto build © Use workspace settings	Source Common So SVD Path
lter matched 11 of 14 items		Re <u>v</u> ert Appl <u>v</u>

2. Select "Debugger" to configure the J-Link and GDB options of the

current debugging project, as shown in Figure 2-17.

For example:

- Device Name: Cortex-M3
- Endianness: Little
- Connection: USB
- Interface: JTAG
- Connect to running target: Close
- Initial speed: Fixed 1000KHz
- GDB port: 2331
- SWO port: 2332
- Telnet: 2333
- Verify downloads: Open
- Initialize registers on start: Open
- Local host only: Open
- Other options: -singlerun -strict -timeout 0 -nogui
- Command: set mem inaccessible-by-default off

Gowin\_EMPU(GW1NS-4C) supports JTAG debug interface. SWD debug interface is not supported.

"Connect to running target" option is disabled.

8 88	8 I	
🗱 Debug Configurations		×
Create, manage, and run configurations		1×
	-	No.
tune filter text		
C/C++ Application	Main % Debugger Startup % Source Common % SVD Path	^
C/C++ Attach to Application	Start the J-Link GDB server locally	
C/C++ Postmortem Debugger	Executable path: \${ilink path}/\${ilink gdbserver}	ariable
C GDB Hardware Debugging	Actual executable: C:/Program Files (x86)/SEGGER/II ink/II inkGDBServerCL exe	=
GDB OpenOCD Debugging	(to change it use the global or workspace preferences pages or the project propert	ties p
GDB PyOCD Debugging	Device name: Cortex-M3 Supported devi	rice na
<ul> <li>C GDB GEMO Debugging</li> <li>✓ C GDB SEGGER J-Link Debugging</li> </ul>	Endianness:	
c cm3_demo Debug	Connection:      USB O IP     (USB serial or IP name/address	ss)
Launch Group	Interface: O SWD () JTAG	
	Initial speed: O Auto O Adaptive  Fixed 1000 kHz	
	GDB port: 2331	
	SWO port: 2332 ☑ Verify downloads ☑ Initialize registers	s on s
	Telnet port: 2333	
	Log file:	Brows
	Other options: -singlerun -strict -timeout 0 -nogui	
	Allocate console for the GDB server	
	GDB Client Setup	
	Executable name: \${cross_prefix}gdb\${cross_suffix} Browse Va	ariable
	Actual executable: arm-none-eabi-gdb	
	Other options:	_
	Commands: set mem inaccessible-by-default off	=
	<	>
Filter matched 11 of 14 items	Re <u>v</u> ert App	ply
(?)	<u>D</u> ebug Cl	lose

#### Figure 2-17 Debugger Configuration Option

#### MCU JTAG Mode Switch

Switch JTAG mode from the download to debug mode using Programmer.

Click "Run> Programmer" on the GMD menu bar or Programmer "##" on the tool bar to open Programmer as shown in 2-11.

Click "Edit > Configure Device" on the menu bar or "Configure Device" ( $\implies$ ) on the tool bar to open the "Device configuration".

If FPGA product GW1NS-4C/GW1NSR-4C is selected, MCU JTAG mode configuration options are as shown in Figure 2-18.

- Select "MCU Mode" in "Access Mode" drop-down list.
- Select "Connect to JTAG of MCU" in "Operation" drop-down list.

Uevice configuration		? >	×
Device Operation			
Access Mode:	MCV Mode	~	
Operation:	Connect to JTAG of MCU	~	
Contract to JINO OF MU			1.1

Figure 2-18 MCU JTAG Mode Configuration for GW1NS-4C/GW1NSR-4C

If FPGA product GW1NSER-4C is selected, MCU JTAG mode configuration options are as shown in Figure 2-19.

- Select "SecureFPGA Mode" in "Access Mode" drop-down list.
- Select "Connect to JTAG of MCU" in "Operation" drop-down list.

Figure 2-19 MCU JTAG Mode Configuration for GW1NSER-4C

🙀 Device configuration			?	×
Device Operation				
Access Mode:	SecureFPGA Mode			$\sim$
Operation:	Connect to JTAG of MCU			$\sim$
Connect to JTAG of M	NCV.			
-		Save	Canc	el

• Click "Save" to complete the configuration.

After device configuration, click "Program/Configure" () in the Programmer tool bar to complete MCU JTAG mode switch.

## MCU JTAG Interface Switch

Taking DK-START-GW1NSR4C-QN48G V1.1 for an instance, switch SW3, SW4, SW5, SW6 (TCK, TDO, TDI, and TMS of JTAG interface) on the development board from "FT232" (Download to "J-LINK" (debug).

## Note!

• The development board must be kept on power in MCU JTAG mode and interface

switch.

 If the development board powers off in the process, it will restore to MCU JTAG download mode automatically after power on.

## **Start Software Debug**

Connect J-LINK emulator.

You can perform operations of breakpoint setting, single-step debug, reset, run, etc. by clicking Debug " \* , as shown in Figure 2-20.

#### Figure 2-20 Start Software Debug

workspace_cm3_hard - Debug - cm3_demo/src/project/led/	led_demo.c - GOW	IN MCU Design	er			-	- C		×
File Edit Source Refactor Navigate Search Project Run Window Help									
[□ * 圖 웹] 團: ♥    ♥ = 히 전 전 10       ♥ 전    ②   ● (4) 중 * ( ○ * ( ) ) ))))))))))									L.
						Quick Acces	s   E		-Xie
∜r Debug ⊠	; <b>i</b> → ▽ □ □	(x)= Variables	🛛 💁 Break	kp 1919 Reg	iste 🚡 P	eriphe 🛋 N	/lodules		
✓	^					#_ ⇒t	ele	3 🖻	$\bigtriangledown$
✓ ♀ cm3_demo.elf		Name		Туре		Value			
<ul> <li>Ihread #1 5/005 (Suspended : Step)</li> <li>Ied demo() at led demo c:29 0xc8</li> </ul>									
main() at main.c:52 0xfc									
JLinkGDBServerCL.exe									~
📓 arm-none-eabi-gdb		<							>
R lad dama c <sup>12</sup> R main c								_	
21				<u>^</u>			S a	<u>ж</u>	~
22 22 (* Definitions */					- a	demo.h		тк	
23 /* Definitions: */ 24					- <b>N</b>	gpio.h			
25 //Application entry function						delay.h			
27 {					۰	led_demo(voi	id) : int		
28 SystemInit(); //Initializes system clock									
30 delay_init(); //Initializes delay functions									
31 32 while(1)									
				~					
<pre>4 GPIO RESETBIT(GPIO0.GPIO PIN 0): //LEDI 0 </pre>	on			>					
📮 Console 🕴 🖉 Tasks 💦 Problems 🜔 Executables 🏮 M	emory			× 🔆 🖹	a: 🖻 🗧	J 🛃 🛃	- 📬	• -	
cm3_demo Debug [GDB SEGGER J-Link Debugging] JLinkGDBSe	rverCL.exe								
Read 4 bytes @ address 0x000000C8 (Data = 0xF81CF000) Removing breakpoint @ address 0x000000C8, Size = 2									^
Reading 64 bytes @ address 0x20003FC0									
Read 4 bytes @ address 0x000000rt (Data = 0x46182300)									~
<									>
	Writable	Smart Insert	29:1	1					6

## 2.3 Reference Design

Gowin\_EMPU(GW1NS-4C) supports reference design in GMD (tested software version V1.2), and you can click <u>here</u> to get the following reference design:

...\ref\_design\MCU\_RefDesign\GMD\_RefDesign\cm3\_demo、 cm3\_freertos、cm3\_rtthread\_nano、cm3\_ucos\_iii

