Gowin PicoRV32 Hardware Design

Reference Manual

IPUG914-1.8E, 09/26/2025

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
01/06/2020	1.0E	Initial version published.
03/12/2020	1.1E	● MCU supports GPIO of Wishbone bus interface. ● MCU supports extension AHB bus interface. ● MCU supports off-chip SPI-Flash download and startup. ● MCU supports the read, write and erasure SPI-Flash. ● MCU supports Hardware Stack Protection and Trap Stack Overflow.
06/01/2020	1.2E	● MCU on-line debug function supported. ● MCU core interrupt handler function enhanced. ● MCU core instruction optimized.
07/16/2021	1.3E	● The synthesis tool "SynplifyPro" removed. ● The version of FPGA software updated.
02/11/2022	1.4E	The range of ITCM and DTCM Size for GW2AN-9X/GW2AN-18X modified.
08/18/2023	1.5E	Arora Ⅴ FPGA products supported.
03/29/2024	1.6E	● GW5AT-60 Version A FPGA products supported. ● Hardware reference design updated.
06/14/2024	1.7E	● GW5ART-15 Version A FPGA products supported. ● Hardware reference design updated.
09/26/2025	1.8E	● Gowin Software version information updated. ● Hardware reference design added.

Contents

List of Figures

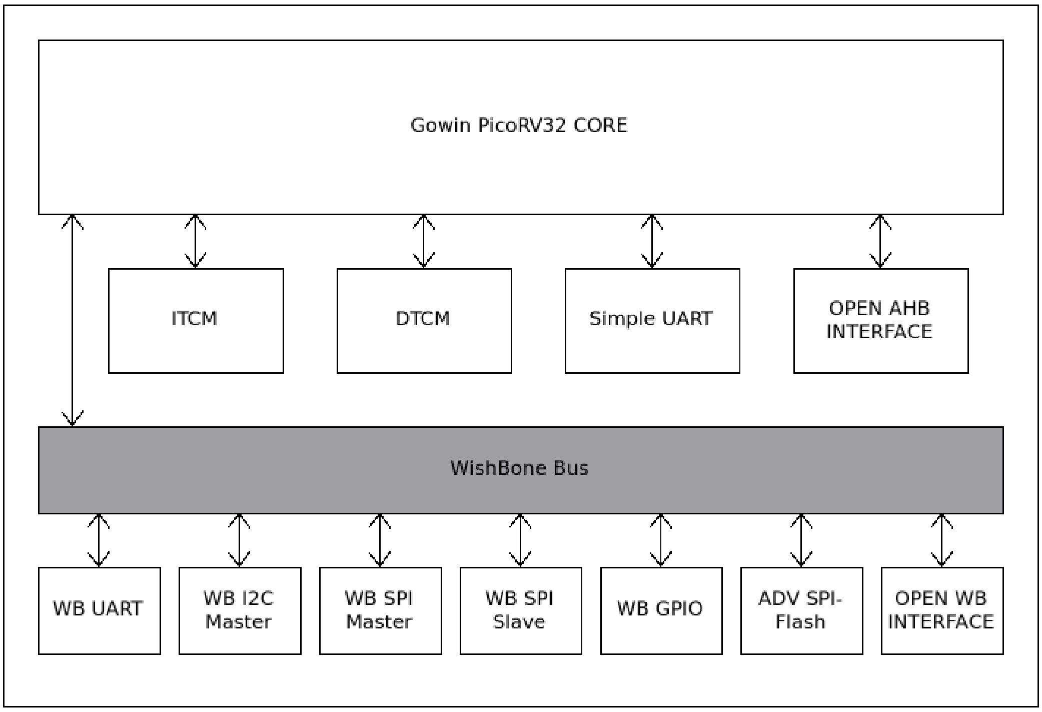
List of Tables

1 Hardware Architecture

1.1 System Architecture

Gowin_PicoRV32 includes PicoRV32 core, instruction memory ITCM, data memory DTCM, simple UART, AHB bus extension interface, Wishbone bus and peripherals, as shown in Figure 1-1.

Figure 1-1 System Architecture



- Gowin PicoRV32 CORE is a microcontroller core with 32-bit RISC-V instruction architecture.
- ITCM is instruction memory.
- DTCM is data memory.
- Simple UART is a configurable simple UART.
- OPEN AHB INTERFACE is the AHB bus extension interface for users to extend peripherals.
- Wishbone Bus connects PicoRV32 Core and peripherals of Wishbone

Bus interface which includes UART, I2C Master, SPI Master, SPI Slave, GPIO, ADV SPI-Flash Memory and Wishbone Bus extension interfaces.

1.2 System Feature

Gowin_PicoRV32 includes two sub-systems:

- The PicoRV32 core subsystem includes the microcontroller core, instruction memory, data memory, a configurable simple UART and AHB bus extension interface.
- Wishbone bus and peripheral subsystem of Wishbone bus interface.

1.2.1 PicoRV32 Core Subsystem

Processor Core

- Risc-V architecture of 32-bit integer instruction set.
- Configurable RISC-V32M multiplication/division instruction set extension and configurable RISC-V32C compression instruction set extension.
- Configurable SPI FLASH Memory interface supporting off-chip SPI-Flash Memory downloading and startup.
- Built-in interrupt processor module with customized interrupt management instructions, supporting 32 interrupt source management. Interrupt priority can be controlled by software.
- Built-in 32-bit timer module with custom timer operation instructions;
- Built-in debug module supporting on-line debug function.
- Big/low endian formats: RISC-V standard only supports low-endian.
- Supports Trap Stack Overflow.

Memory

- ITCM: instruction memory. The size can be configured as 8/16/32/64/128/256/512KB. The data and instruction is low endian.
- DTCM: Data memory. The size can be configured as 8/16/32/64/128/256/512KB. The data and instruction is low endian. It supports Hardware Stack Protection.

Simple UART

- Simple UART, serial communication interface.
- Use very few logical resources.

OPEN AHB INTERFACE

- AHB bus extension interface
- Users can extend to connect customized AHB bus interface peripherals.

1.2.2 Wishbone Bus Sub-system

Wishbone Bus subsystem includes UART, I2C Master, SPI Master, SPI Slave, GPIO, ADV SPI Flash Memory and Wishbone bus extension interfaces.

1.3 System Port Definition

The definition of Gowin_PicoRV32 ports is as shown in Table 1-1.

Table 1-1 Definition of System Ports

Name	I/O	Data Width	Description	Module
clk_in	in	1	System clock signal	-
resetn_in	in	1	System reset signal	-
irq_in	in	12	External interrupt input signal	OPEN WB INTERFACE and OPEN AHB INTERFACE
jtag_TDI	in	1	JTAG data input signal	Debug
jtag_TCK	in	1	JTAG clock input signal	
jtag_TMS	in	1	JTAG mode selection signal	
jtag_TDO	out	1	JTAG data output signal	
ser_tx	out	1	Output signal of Simple UART	Simple UART
ser_rx	in	1	Input signal of Simple UART	
gpio_io	inout	32	Input and output signal of Wishbone GPIO	WB GPIO
wbuart_tx	out	1	Output signal of Wishbone UART	WB UART
wbuart_rx	in	1	Input signal of Wishbone UART	
wbi2c_sda	inout	1	Data signal of Wishbone I2C Master	WB I2C Master
wbi2c_scl	inout	1	Clock signal of Wishbone I2C Master	
wbspi_master_miso	in	1	MISO signal of Wishbone SPI Master	WB SPI Master
wbspi_master_mosi	out	1	MISO signal of Wishbone SPI Master	
wbspi_master_ssn	out		SLAVE selected signal of Wishbone SPI Master Each Slave corresponds to 1 bit, up to 8 bits	
wbspi_master_sclk	out	1	Clock signal of Wishbone SPI Master	
wbspi_slave_miso	out	1	MISO signal of Wishbone SPI Slave	WB SPI Slave
wbspi_slave_	in	1	MISO signal of Wishbone SPI	

Name	I/O	Data Width	Description	Module
mosi			Slave	
wbspi_slave_ssn	in	1	SLAVE selected signal of Wishbone SPI Slave	
wbspi_slave_sclk	in	1	Clock signal of Wishbone SPI Slave	
io_spi_clk	inout	1	Clock signal of ADV SPI-Flash	ADV SPI-Flash Memory
io_spi_csn	inout	1	Chip selected signal of ADV SPI-Flash	
io_spi_mosi	inout	1	MOSI signal of ADV SPI-Flash	
io_spi_miso	inout	1	MISO signal of ADV SPI-Flash	
slv_ext_stb_o	out	1	strb signal of Wishbone bus extension interface	OPEN WB INTERFACE
slv_ext_we_o	out	1	Write operation signal of Wishbone bus extension interface	
slv_ext_cyc_o	out	1	cyc signal of Wishbone bus extension interface	
slv_ext_ack_i	in	1	ack signal of Wishbone bus extension interface	
slv_ext_adr_o	out	32	Address signal of Wishbone bus extension interface	
slv_ext_wdata_o	out	32	Write data signal of Wishbone bus extension interface	
slv_ext_rdata_i	in	32	Read data signal of Wishbone bus extension interface	
slv_ext_sel_o	out	4	Byte selection signal of Wishbone bus extension interface	
hrdata	in	32	Read data signal of AHB bus extension interface	OPEN AHB INTERFACE
hresp	in	2	Bus transmission status signal of AHB bus extension interface	
hready	in	1	Ready signal of AHB bus extension interface	
haddr	out	32	Address signal of AHB bus extension interface	
hwrite	out	1	Read and Write I/O signal of AHB bus extension interface	
hsize	out	3	Transmission data size signal of AHB bus extension interface	
hburst	out	3	Transmission Burst signal of AHB bus extension interface	
hwdata	out	32	Write data signal of AHB bus extension interface	

Name	I/O	Data Width	Description	Module
hsel	out	1	Chip selected signal of AHB bus extension interface	
htrans	out	2	Transmission Type signal of AHB bus extension interface	

1.4 System Resource Statistics

Taking GW5A-25 Version A as an example, the system resource statistics of Gowin_PicoRV32 is as shown in Table 1-2.

Table 1-2 System Resource Statistics

Resources / Configuration	LUT	Register	BSRAM	DSP
MCU core	5937	3583	32	2
MCU core+ Bus peripherals	8542	5405	32	2

2 Hardware Design Flow

2.1 Hardware Target

- DK_START_GW2A18 V2.0
  GW2A-LV18PG256C8/I7
  GW2A-18 (Version C)
- DK_START_GW2A-LV55PG484C8I7 V1.3
  GW2A-LV55PG484C8/I7
  GW2A-55 (Version C)
- DK_START_GW5AST-LV138FPG676A V1.0
  GW5AST-LV138FPG676AC1/I0
  GW5AST-138 (Version B)
- DK_DP_GW5AT-LV60UG225 V1.0
  GW5AT-LV60UG225C2/I1
  GW5AT-60 (Version B)
- DK_START_GW5A-LV25UG324 V2.0
  GW5A-LV25UG324C2/I1
  GW5A-25 (Version A)

2.2 Software Version

Tested software version: Gowin_V1.9.11.02 (64-bit)

2.3 Softcore Generator

Gowin Software provides a softcore generator, IP Core Generator. It can be used to configure and generate Gowin_PicoRV32 IP design.

2.4 Download Software

Gowin_PicoRV32 supports the download of the bitstream files in hardware design with the download tool, Programmer.

For the usage of Gowin Programmer, please see *SUG502, Gowin Programmer User Guide*.

2.5 Design Flow

Gowin_PicoRV32 hardware design flow is as follows:

1. Configure the PicoRV32 core subsystem and Wishbone Bus subsystem in IP Core Generator tool to generate the Gowin_PicoRV32 IP design; import the project.

Note!

If you need to support customized peripherals through Wishbone bus extension interface, enable OPEN WB INTERFACE or OPEN AHB INTERFACE, and disable "Use Gowin PicoRV32 as top module".

2. Instantiate Gowin_PicoRV32 IP Top Module, import user design, and connect user designs with Gowin_PicoRV32 IP Top Module.
3. Add physical and timing Constraints.
4. Use GowinSynthesis to synthesis and generate the netlist file.
5. Run Place & Route to generate the bitstream files in hardware design.
6. Use Programmer to download the bitstream files in hardware design to chip.

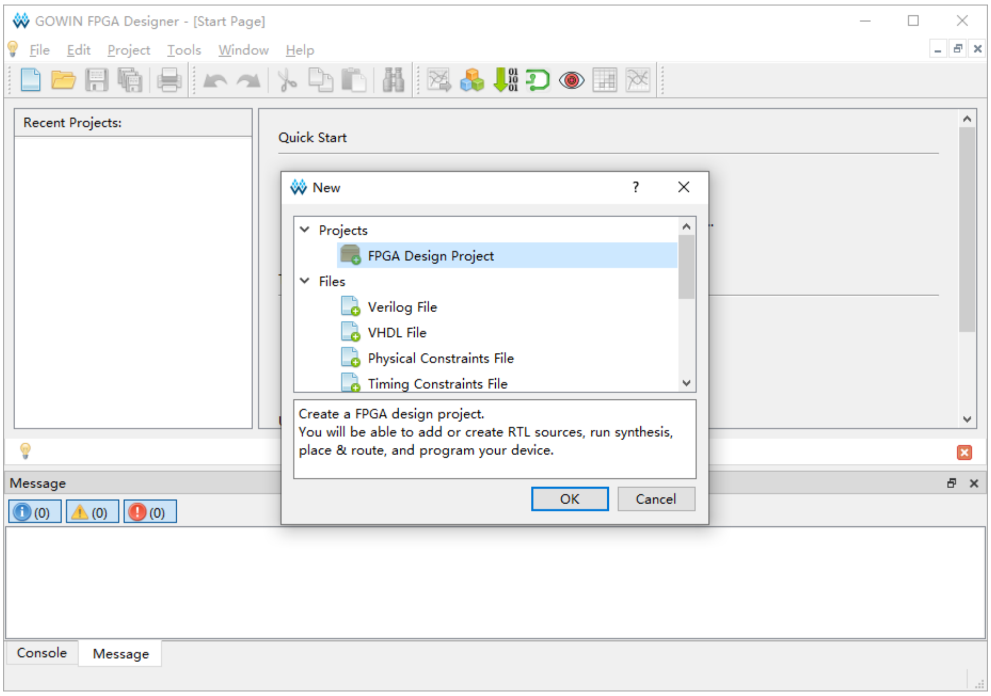
3 Project Template

3.1 Project Creation

3.1.1 Create a New Project

Double-click to open Gowin Software. Select "File > New… > FPGA Design Project" on the menu bar, or click "🗋" or "Quick Start" on tool bar to create an FPGA Design project, as shown in Figure 3-1.

Figure 3-1 New Projects

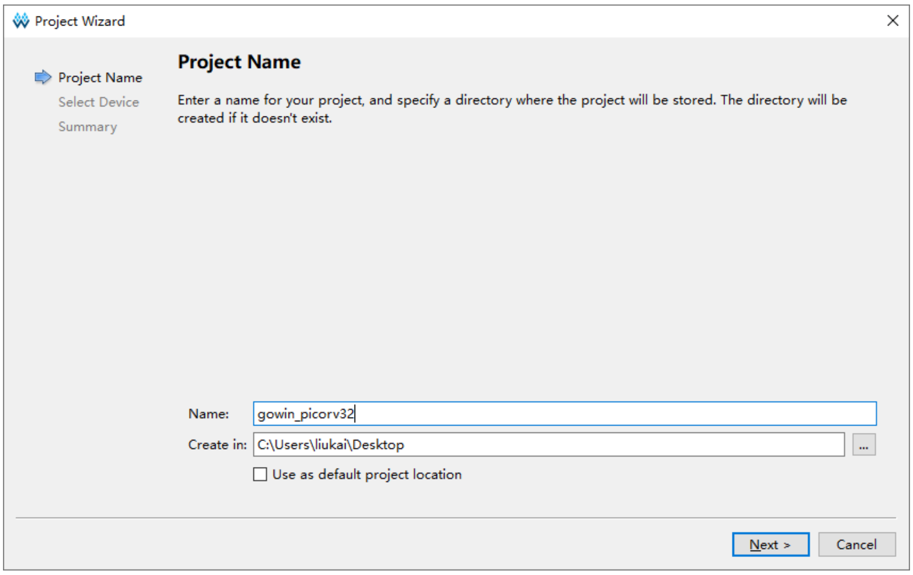


Create a FPGA design project.
You will be able to add or create RTL sources, run synthesis, place & route, and program your device.

3.1.2 Set Project Name and Path

Enter the project name and select the project path, as shown in Figure 3-2.

Figure 3-2 Set Project Name and Path



Project Name

Enter a name for your project, and specify a directory where the project will be stored. The directory will be created if it doesn't exist.

Name: gowin_picorv32

Create in: C:\Users\liukai\Desktop

□ Use as default project location

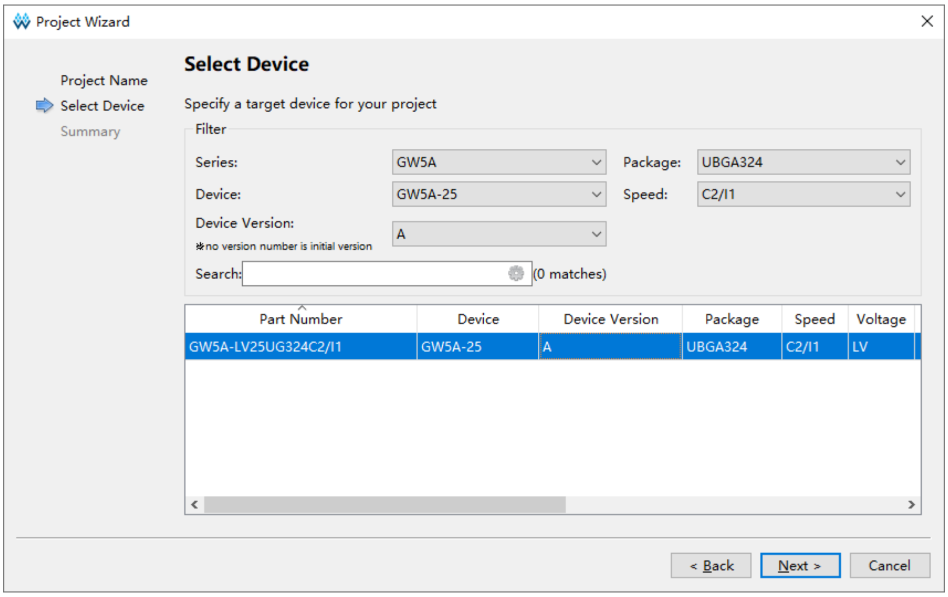
3.1.3 Select Device

Select "Series", "Device", "Device Version", "Package", "Speed", and "Part Number", as shown in Figure 3-3.

Take reference design for DK_START_GW5A-LV25UG324 V2.0 as an example.

- Series: GW5A
- Device: GW5A-25
- Device Version: A
- Package: UBGA324
- Speed: C2/I1
- Part Number: GW5A-LV25UG324C2/I1

Figure 3-3 Select Device



Select Device

Specify a target device for your project

Filter

Series: GW5A

Package: UBGA324

Device: GW5A-25

Speed: C2/I1

Device Version: A

✱no version number is initial version

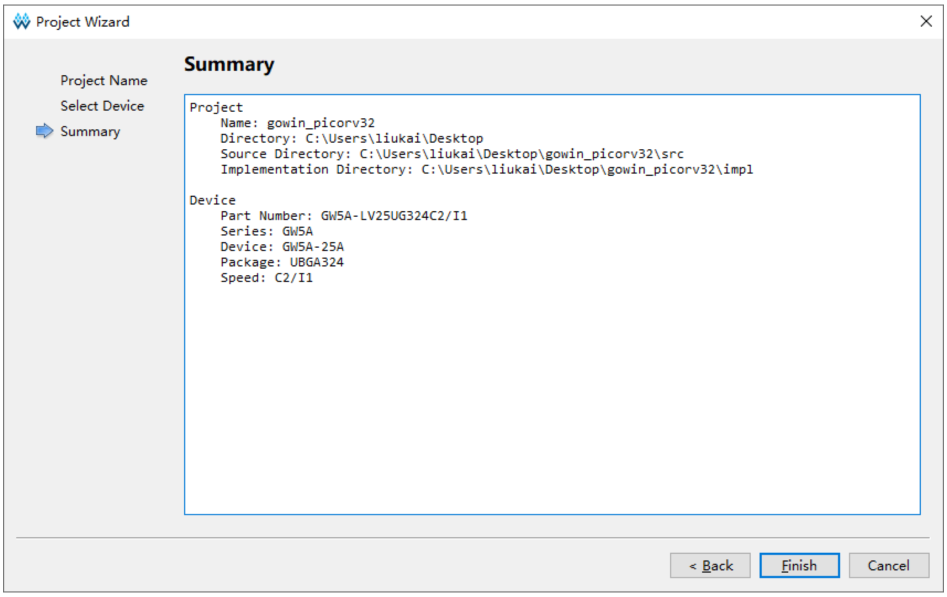
Search: (0 matches)

Part Number	Device	Device Version	Package	Speed	Voltage
GW5A-LV25UG324C2/I1	GW5A-25	A	UBGA324	C2/I1	LV

3.1.4 Complete Project Creation

Complete the creation of the new project, as shown in Figure 3-4.

Figure 3-4 Complete Project Creation



```
Project
    Name: gowin_picorv32
    Directory: C:\Users\liukai\Desktop
    Source Directory: C:\Users\liukai\Desktop\gowin_picorv32\src
    Implementation Directory: C:\Users\liukai\Desktop\gowin_picorv32\impl

Device
    Part Number: GW5A-LV25UG324C2/I1
    Series: GW5A
    Device: GW5A-25A
    Package: UBGA324
    Speed: C2/I1
```

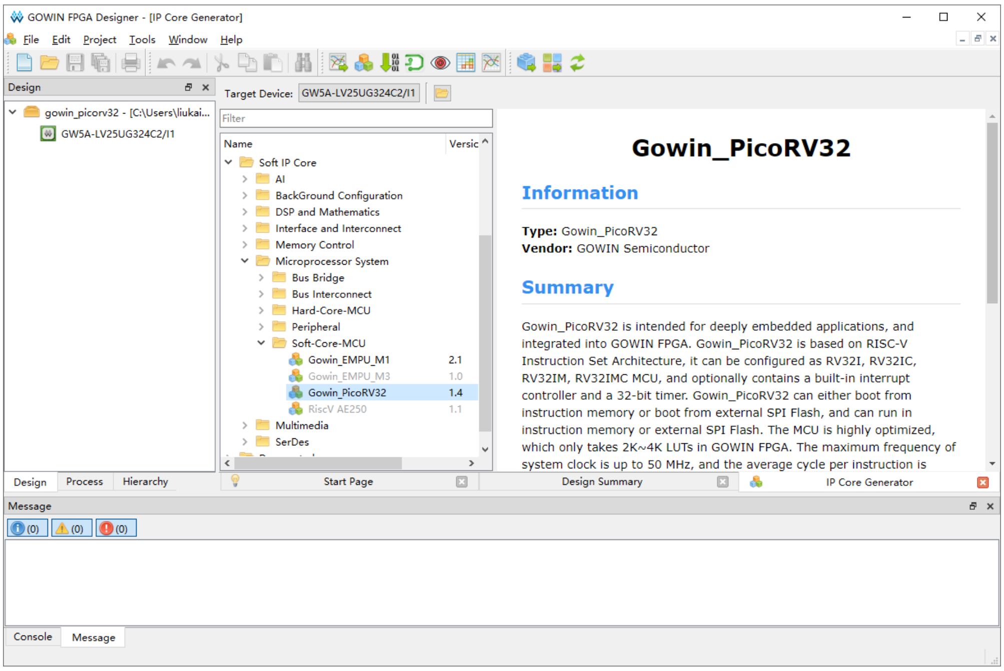
3.2 Hardware Design

Use IP Core Generator to generate Gowin_PicoRV32 IP design.

Select "Tools > IP Core Generator" in the menu bar or " " in the tool bar to open the IP Core Generator.

Select "Soft IP Core > Microprocessor System > Soft-Core-MCU > Gowin_PicoRV32 1.4", as shown in Figure 3-5.

Figure 3-5 Select Gowin_PicoRV32



Gowin_PicoRV32

Information

Type: Gowin_PicoRV32
Vendor: GOWIN Semiconductor

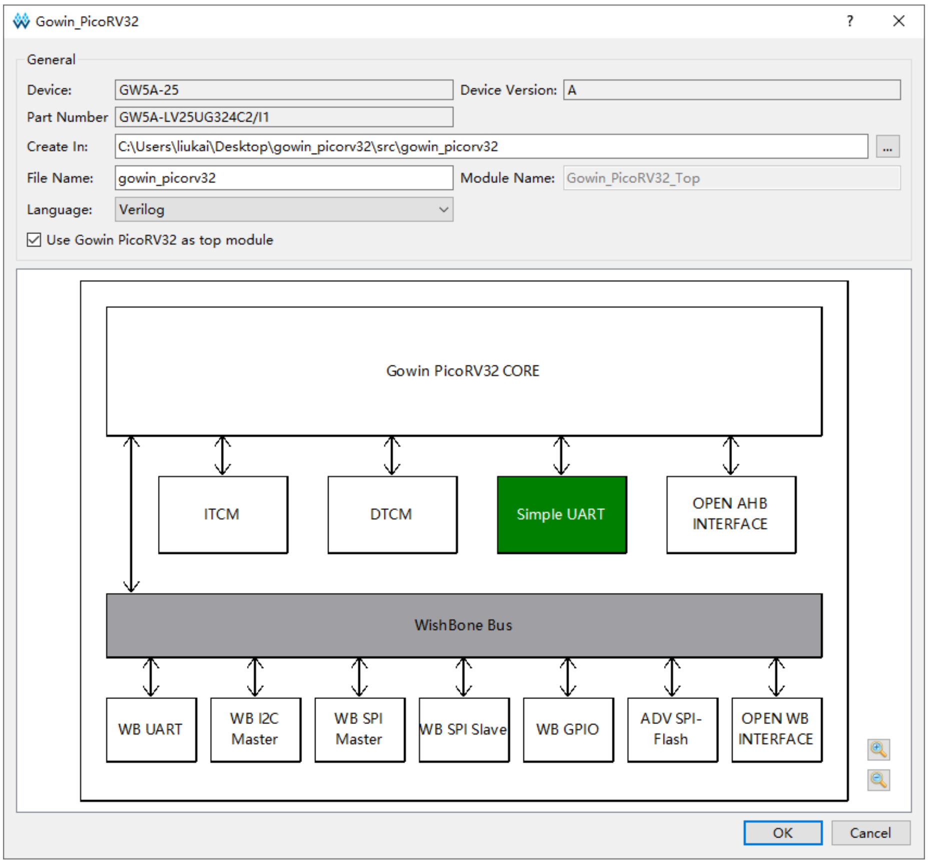
Summary

Gowin_PicoRV32 is intended for deeply embedded applications, and integrated into GOWIN FPGA. Gowin_PicoRV32 is based on RISC-V Instruction Set Architecture, it can be configured as RV32I, RV32IC, RV32IM, RV32IMC MCU, and optionally contains a built-in interrupt controller and a 32-bit timer. Gowin_PicoRV32 can either boot from instruction memory or boot from external SPI Flash, and can run in instruction memory or external SPI Flash. The MCU is highly optimized, which only takes 2K~4K LUTs in GOWIN FPGA. The maximum frequency of system clock is up to 50 MHz, and the average cycle per instruction is

Double-click to open Gowin_PicoRV32 including the Gowin PicoRV32 core subsystem and Wishbone Bus subsystem. Gowin_PicoRV32 configuration options are as shown in Figure 3-6.

If the module configuration is enabled, the module is green.

Figure 3-6 Gowin_PicoRV32 Configuration Option



Gowin_PicoRV32

General

Device: GW5A-25

Device Version: A

Part Number: GW5A-LV25UG324C2/I1

Create In: C:\Users\liukai\Desktop\gowin_picorv32\src\gowin_picorv32

File Name: gowin_picorv32

Module Name: Gowin_PicoRV32_Top

Language: Verilog

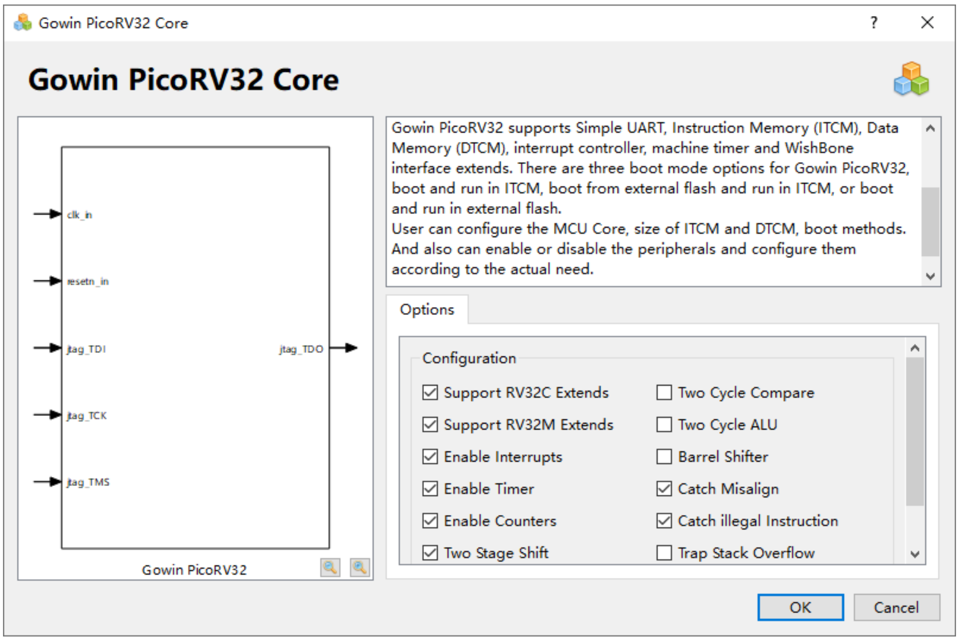
☑ Use Gowin PicoRV32 as top module

3.2.1 PicoRV32 Core Subsystem Configuration

Gowin PicoRV32 IP CORE Configuration

Double-click Gowin PicoRV32 CORE to open the configuration view of Gowin PicoRV32 CORE, as shown in Figure 3-7.

Figure 3-7 Gowin PicoRV32 CORE Configuration Option



Gowin PicoRV32 Core

Gowin PicoRV32 supports Simple UART, Instruction Memory (ITCM), Data Memory (DTCM), interrupt controller, machine timer and WishBone interface extends. There are three boot mode options for Gowin PicoRV32, boot and run in ITCM, boot from external flash and run in ITCM, or boot and run in external flash.
User can configure the MCU Core, size of ITCM and DTCM, boot methods. And also can enable or disable the peripherals and configure them according to the actual need.

Configuration

☑ Support RV32C Extends
☑ Support RV32M Extends
☑ Enable Interrupts
☑ Enable Timer
☑ Enable Counters
☑ Two Stage Shift
□ Two Cycle Compare
□ Two Cycle ALU
□ Barrel Shifter
☑ Catch Misalign
☑ Catch illegal Instruction
□ Trap Stack Overflow

Gowin PicoRV32 CORE configuration is as shown in Table 3-1.

Table 3-1 Gowin PicoRV32 CORE Configuration Option

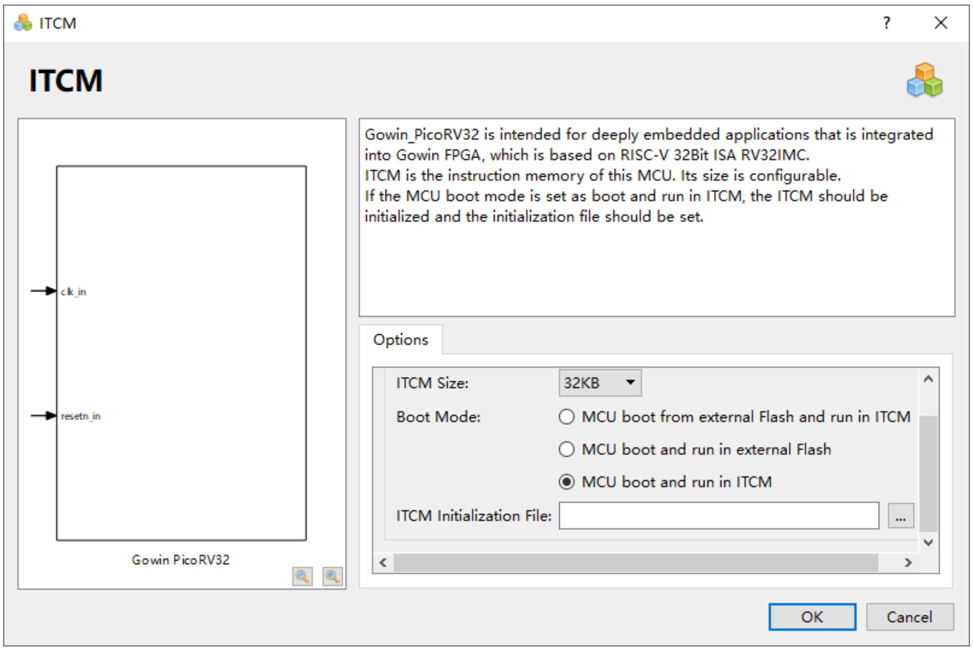
Options	Description
Support RV32C Extends	Select whether to support RISC-V compression instruction set extension, selected by default.
Support RV32M Extends	Select whether to support RISC-V multiplication/division instruction set extension, selected by default.
Enable Interrupts	Select whether to support interrupt control, selected by default.
Enable Timer	Select whether to support timer, selected by default.
Enable Counters	Select whether to support counter instruction of RDCYCLE[H]/RDTIME[H]/RDINSTRET[H], selected by default.
Two Stage Shift	Select whether to support two stage shift (if so, you can speed up the shift operation, but the logical resource usage will increase), selected by default.
Two Cycle Compare	Select whether to support two cycle Compare. (if so, you can shorten the length of the data path and improve timing quality, but the Compare instruction is executed, thus adding one clock cycle, disabled by default.
Two Cycle ALU	Choose whether to support two cycle ALU. (if so, you can shorten the length of the data path and improve timing quality, but the ALU instruction is executed, thus adding one clock cycle, disabled by default.
Barrel Shifter	Select whether to support Barrel Shifter, disabled by default.
Catch Misalign	Select whether to enter the TRAP and stop running if address misalignment occurs during memory access, selected by default.

Options	Description
Catch illegal Instruction	Select whether to enter the TRAP and stop running while executing an illegal instruction, selected by default.
Trap Stack Overflow	Select whether to support trap stack overflow, disabled by default.
Enable Debug	Select whether to support on-line debug, selected by default.

ITCM Configuration

Double-click ITCM to open the configuration view, as shown in Figure 3-8. You can configure ITCM Size, three methods to boot Gowin_PicoRV32, and ITCM initialization files on this view.

Figure 3-8 ITCM Configuration Option



Gowin_PicoRV32 is intended for deeply embedded applications that is integrated into Gowin FPGA, which is based on RISC-V 32Bit ISA RV32IMC.
ITCM is the instruction memory of this MCU. Its size is configurable.
If the MCU boot mode is set as boot and run in ITCM, the ITCM should be initialized and the initialization file should be set.

ITCM Size: 32KB

Boot Mode:
○ MCU boot from external Flash and run in ITCM
○ MCU boot and run in external Flash
● MCU boot and run in ITCM

ITCM Initialization File:

- ITCM Size
  - It can be configured 8KB, 16KB, 32KB, 64KB, 128KB, 256KB, or 512KB.
  - For GW1N/R-9, the maximum ITCM Size is 32KB. The default is 16KB.
  - For GW2AN-9X/18X, the maximum ITCM Size is 32KB. The default is 16KB.
  - For GW2A/R/NR-18, the maximum ITCM Size is 64KB. The default is 32KB.
  - For GW2A/N-55, the maximum ITCM Size is 256KB. The default is 64KB.
  - For GW5A/T/ST/S-138, the maximum ITCM Size is 512KB. The default is 218KB.

  - For GW5AT-75, the maximum ITCM Size is256KB. The default is 128KB.
  - For GW5A/R/S-25, the maximum ITCM Size is 64KB. The default is 32KB.
  - For GW5AT/RT-15, the maximum ITCM Size is 64KB. The default is 32KB.
  - For GW5A/T-60, the maximum ITCM Size is 128KB. The default is 64KB.
- Boot Mode
  - MCU boot from off-chip SPI Flash and run in ITCM
  - MCU boot and run in off-chip SPI Flash
  - MCU boot and run in ITCM

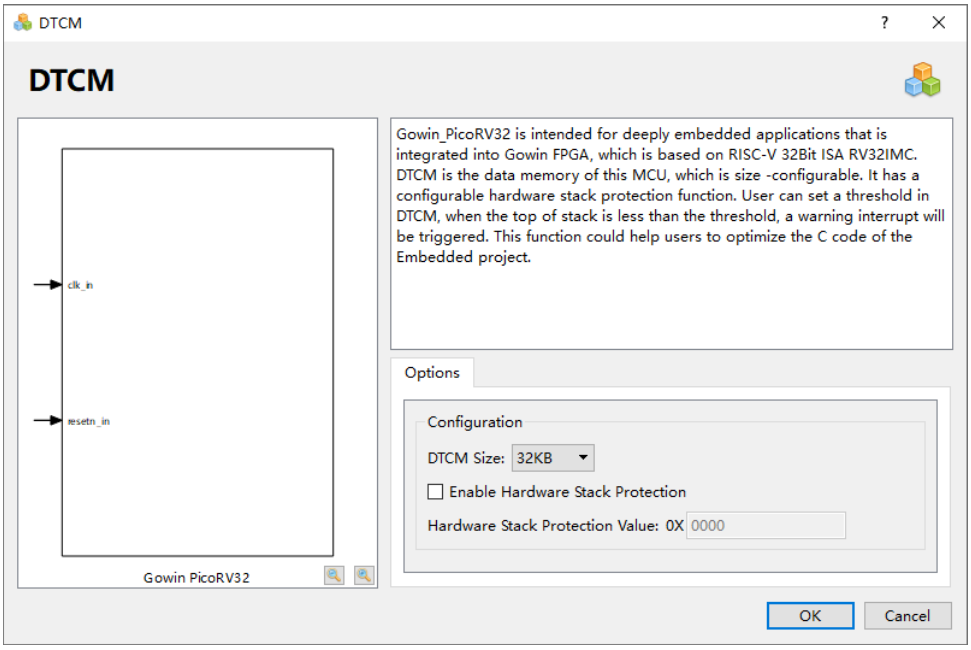
Note!

- If you choose MCU boot and run in ITCM, import the ITCM Initialization File (Gowin_PicoRV32 software programming design ram32.hex) in the ITCM Initialization File.
- The path that ITCM Initialization File is imported cannot contain numbers or escape characters such as \n and \t.

DTCM Configuration

Double-click DTCM to open the DTCM configuration view, as shown in Figure 3-9. You can configure DTCM Size, Hardware Stack Protection, and Hardware Stack Protection Value on this view.

Figure 3-9 DTCM Configuration Option



Gowin_PicoRV32 is intended for deeply embedded applications that is integrated into Gowin FPGA, which is based on RISC-V 32Bit ISA RV32IMC. DTCM is the data memory of this MCU, which is size -configurable. It has a configurable hardware stack protection function. User can set a threshold in DTCM, when the top of stack is less than the threshold, a warning interrupt will be triggered. This function could help users to optimize the C code of the Embedded project.

Configuration

DTCM Size: 32KB

□ Enable Hardware Stack Protection

Hardware Stack Protection Value: 0X 0000

- DTCM Size
  - It can be configured 8KB, 16KB, 32KB, 64KB, 128KB, 256KB or 512KB.
  - For GW1N/R-9, the maximum DTCM Size is 32KB. The default is

16KB.

  - For GW2AN-9X/18X, the maximum DTCM Size is 32KB. The default is 16KB.
  - For GW2A/R/NR-18, the maximum DTCM size is 64KB. The default is 32KB.
  - For GW2A/N-55, the maximum DTCM Size is 256KB. The default is 64KB.
  - For GW5A/T/ST/S-138, the maximum DTCM Size is 512KB. The default is 128KB.
  - For GW5AT-75, the maximum DTCM Size is 256KB. The default is 128KB.
  - For GW5A/R/S-25, the maximum DTCM Size is 64KB. The default is 32KB.
  - For GW5AT/RT-15, the maximum DTCM Size is 64KB. The default is 32KB.
  - For GW5A/T-60, the maximum DTCM Size is 128KB. The default is 64KB.
- Hardware Stack Protection
  - If Enable Hardware Stack Protection is enabled, Gowin_PicoRV32 supports DTCM hardware stack protection.
  - The value of hardware stack protection is smaller than DTCM Size.

ITCM and DTCM Configuration Limitations

- For GW1N/R-9, ITCM or DTCM can be configured up to 32KB. If ITCM or DTCM has been configured to 32KB, the other can only be configured up to 16KB.
- For GW2AN-9X/18X, ITCM or DTCM can be configured up to 32KB. If ITCM or DTCM has been configured to 32KB, the other can only be configured up to 16KB.
- For GW2A/R/NR-18, ITCM or DTCM can be configured up to 64KB. If ITCM or DTCM has been configured to 64KB, the other can only be configured up to 16KB.
- For GW2A/N-55, ITCM or DTCM can be configured up to 256KB. If ITCM or DTCM has been configured to 32KB, the other can only be configured up to 16KB.
- For GW5A/T/ST/S-138, ITCM or DTCM can be configured up to 512KB. If ITCM or DTCM has been configured to 512KB, the other can only be configured up to 128KB.
- For GW5AT-75, ITCM or DTCM can be configured up to 256KB. If ITCM or DTCM has been configured to 256KB, the other can only be configured up to 256KB.
- For GW5A/R/S-25, ITCM or DTCM can be configured up to 64KB. If ITCM or DTCM has been configured to 64KB, the other can only be configured up to 32KB.

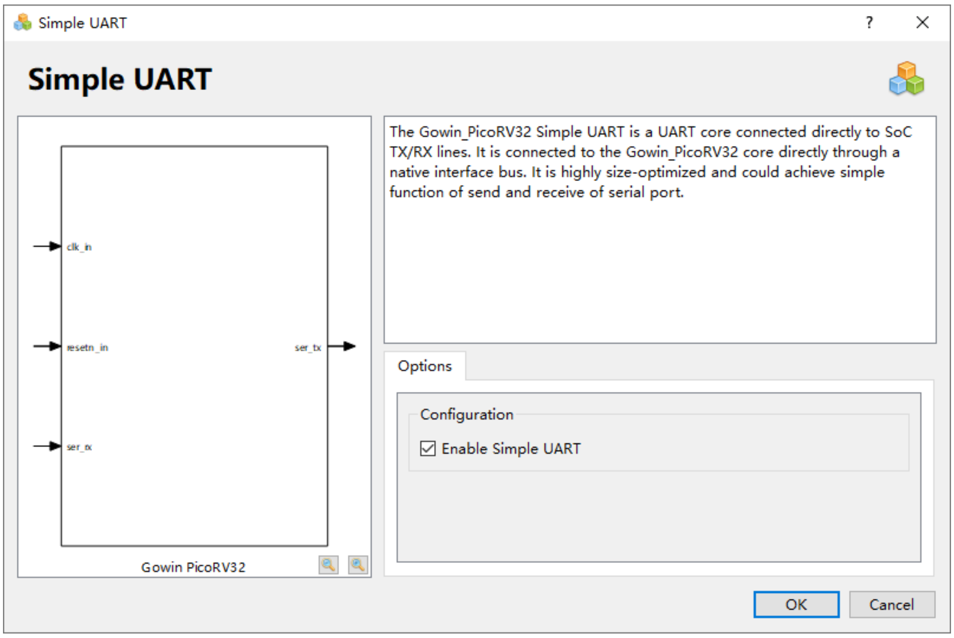
- For GW5AT/RT-15, ITCM or DTCM can be configured up to 64KB. If ITCM or DTCM has been configured to 64KB, the other can only be configured up to 8KB.
- For GW5A/T-60, ITCM or DTCM can be configured up to 128KB. If ITCM or DTCM has been configured to 128KB, the other can only be configured up to 64KB.

Simple UART Configuration

Double-click Simple UART to open the Simple UART configuration view, as shown in Figure 3-10. You can configure whether to enable Simple UART or not.

If "Enable Simple UART" is selected, Gowin_PicoRV32 supports Simple UART. The default is selected.

Figure 3-10 Simple UART Configuration Option



The Gowin_PicoRV32 Simple UART is a UART core connected directly to SoC TX/RX lines. It is connected to the Gowin_PicoRV32 core directly through a native interface bus. It is highly size-optimized and could achieve simple function of send and receive of serial port.

Options

Configuration

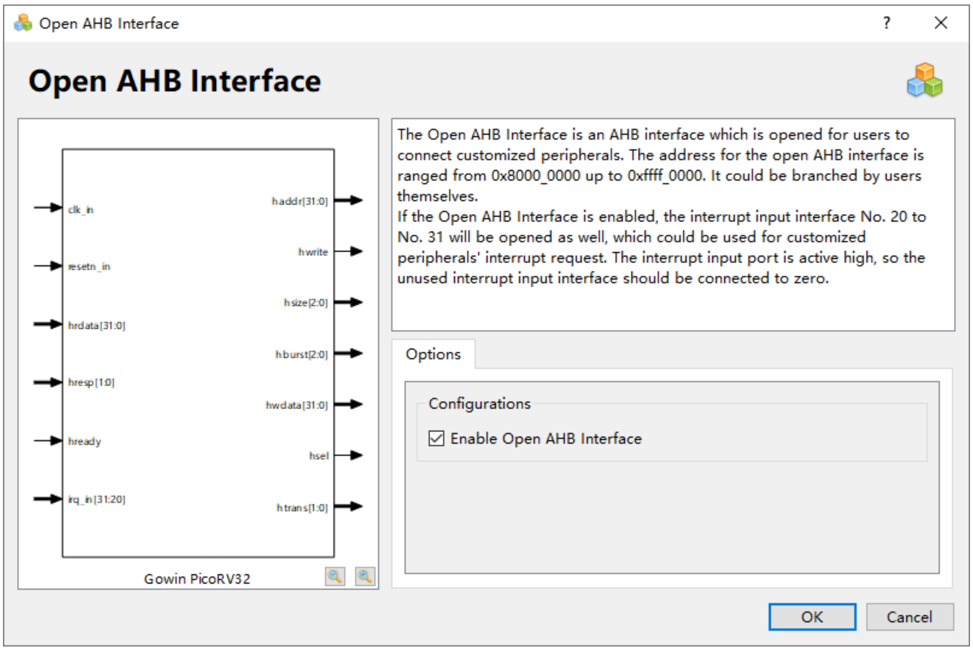
☑ Enable Simple UART

OPEN AHB INTERFACE Configuration Options

Double-click "OPEN AHB INTERFACE" to open the configuration view of "OPEN AHB INTERFACE", as shown in Figure 3-11.

- The default is disabled.
- If "Enable Open AHB Interface" option is selected, Gowin_PicoRV32 supports "OPEN AHB INTERFACE". You can connect peripherals used to extend AHB bus interface on this interface;
- There are 12 external interrupt signals irq_in[31:20] reserved for peripherals.

Figure 3-11 OPEN AHB INTERFACE Configuration Option



The Open AHB Interface is an AHB interface which is opened for users to connect customized peripherals. The address for the open AHB interface is ranged from 0x8000_0000 up to 0xffff_0000. It could be branched by users themselves.
If the Open AHB Interface is enabled, the interrupt input interface No. 20 to No. 31 will be opened as well, which could be used for customized peripherals' interrupt request. The interrupt input port is active high, so the unused interrupt input interface should be connected to zero.

Configurations

☑ Enable Open AHB Interface

3.2.2 Wishbone Bus Sub-system Configuration

Wishbone Bus subsystem can be configured to support peripherals, including WB UART, WB I2C Master, WB SPI Master, WB SPI Slave, WB GPIO, ADV SPI-Flash Memory, and OPEN WB INTERFACE.

The configuration options of Wishbone Bus subsystem is as shown in Table 3-2.

Table 3-2 Wishbone Bus Subsystem Configuration Option

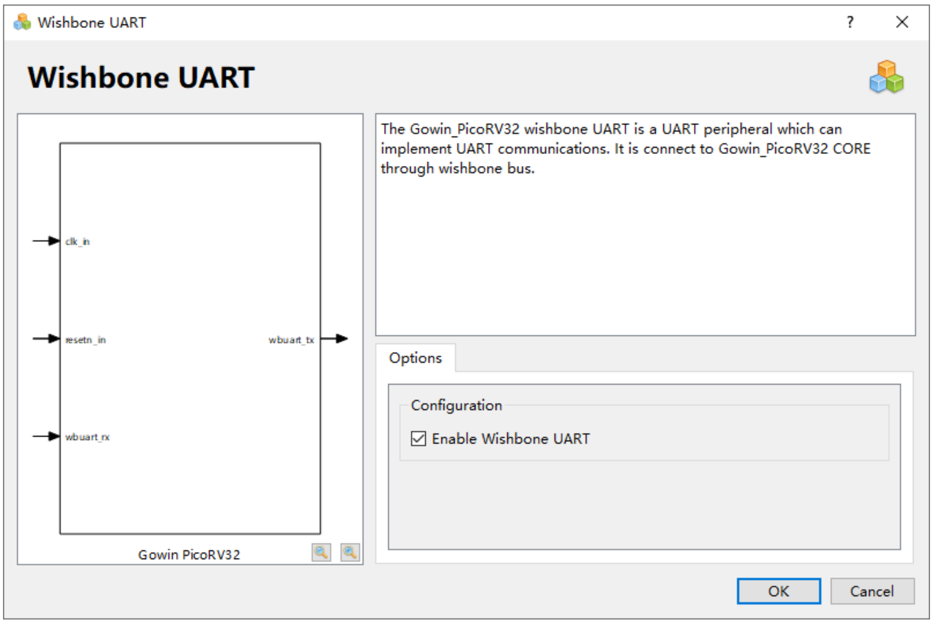
Options	Description
Enable Wishbone UART	Enable WB UART or not, disabled by default
Enable Wishbone I2C Master	Enable WB I2C Master or not, disabled by default
Enable Wishbone SPI Master	Enable WB SPI Master or not, disabled by default
Enable Wishbone SPI Slave	Enable WB SPI Slave or not, disabled by default
Enable Wishbone GPIO	Enable WB GPIO or not, disabled by default
Enable ADV SPI-Flash	Enable ADV SPI-Flash or not, disabled by default
Enable Open Wishbone Interface	Enable OPEN WB INTERFACE or not, disabled by default

WB UART Configuration Options

Double-click WB UART to open the configuration view of Wishbone UART, as shown in Figure 3-12.

If "Enable Wishbone UART" option is selected, Gowin_PicoRV32 supports Wishbone UART, the default is disabled.

Figure 3-12 WB UART Configuration Option



Wishbone UART

The Gowin_PicoRV32 wishbone UART is a UART peripheral which can implement UART communications. It is connect to Gowin_PicoRV32 CORE through wishbone bus.

Options

Configuration

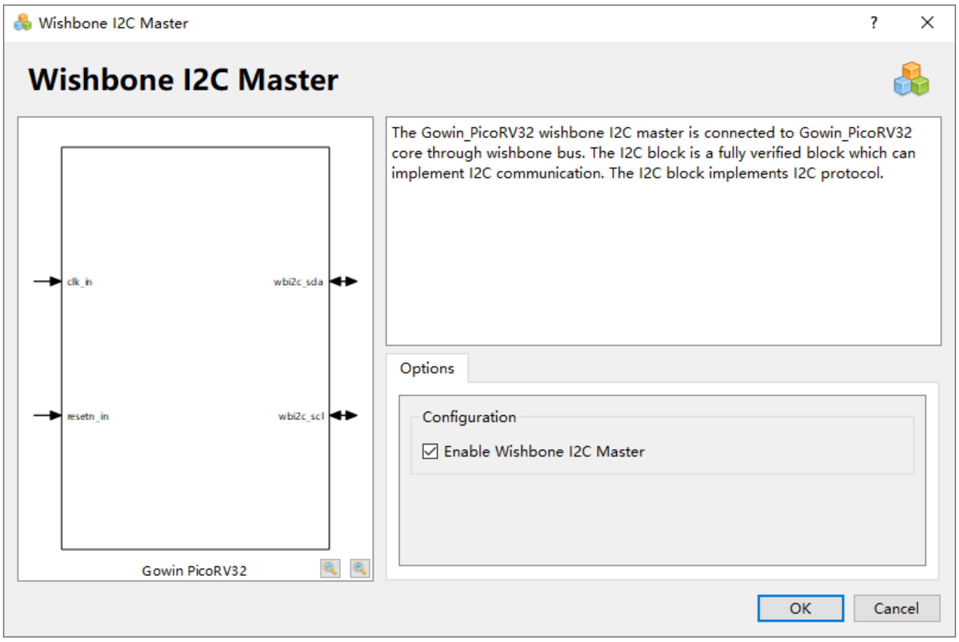
☑ Enable Wishbone UART

WB I2C Master Configuration Options

Double-click WB I2C Master to open the configuration view of Wishbone I2C Master, as shown in Figure 3-13.

If "Enable Wishbone I2C Master" option is selected, Gowin_PicoRV32 supports Wishbone I2C Master. The default is disabled.

Figure 3-13 WB I2C Master Configuration Option



Wishbone I2C Master

The Gowin_PicoRV32 wishbone I2C master is connected to Gowin_PicoRV32 core through wishbone bus. The I2C block is a fully verified block which can implement I2C communication. The I2C block implements I2C protocol.

Options

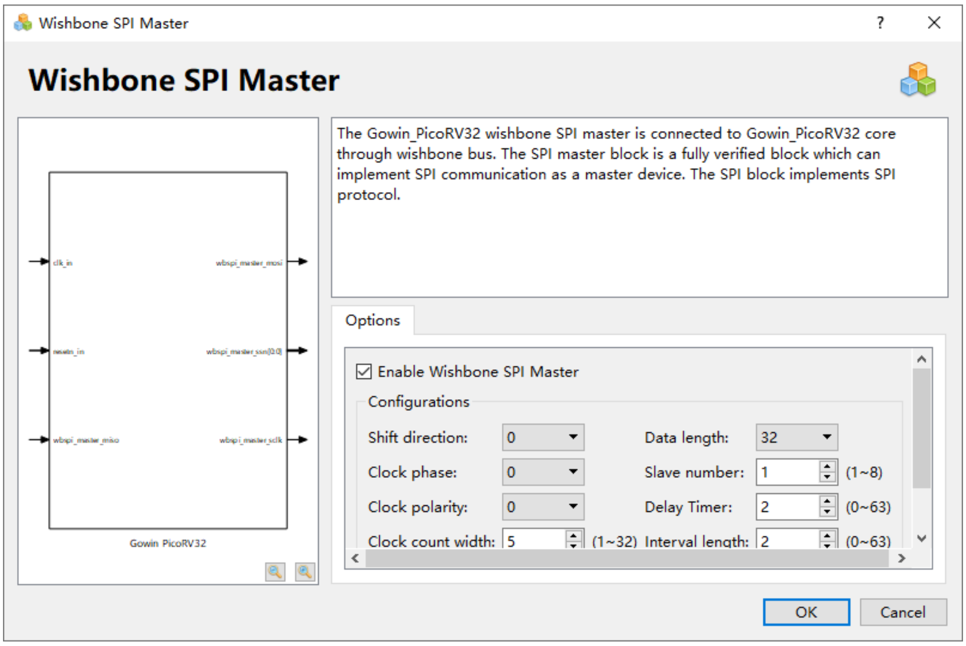
Configuration

☑ Enable Wishbone I2C Master

WB SPI Master Configuration Options

Double-click WB SPI Master to open the configuration view of Wishbone SPI Master, as shown in Figure 3-14.

Figure 3-14 WB SPI Master Configuration Option



Wishbone SPI Master

The Gowin_PicoRV32 wishbone SPI master is connected to Gowin_PicoRV32 core through wishbone bus. The SPI master block is a fully verified block which can implement SPI communication as a master device. The SPI block implements SPI protocol.

☑ Enable Wishbone SPI Master

Configurations

Shift direction: 0 Data length: 32

Clock phase: 0 Slave number: 1 (1~8)

Clock polarity: 0 Delay Timer: 2 (0~63)

Clock count width: 5 (1~32) Interval length: 2 (0~63)

- If "Enable Wishbone SPI Master" option is selected, Gowin_PicoRV32 supports Wishbone SPI Master. The default is disabled.
- If you select "Wishbone SPI Master", Wishbone SPI Master parameters can be configured, as shown in Table 3-3.

Table 3-3 WB SPI Master Parameter Configuration Option

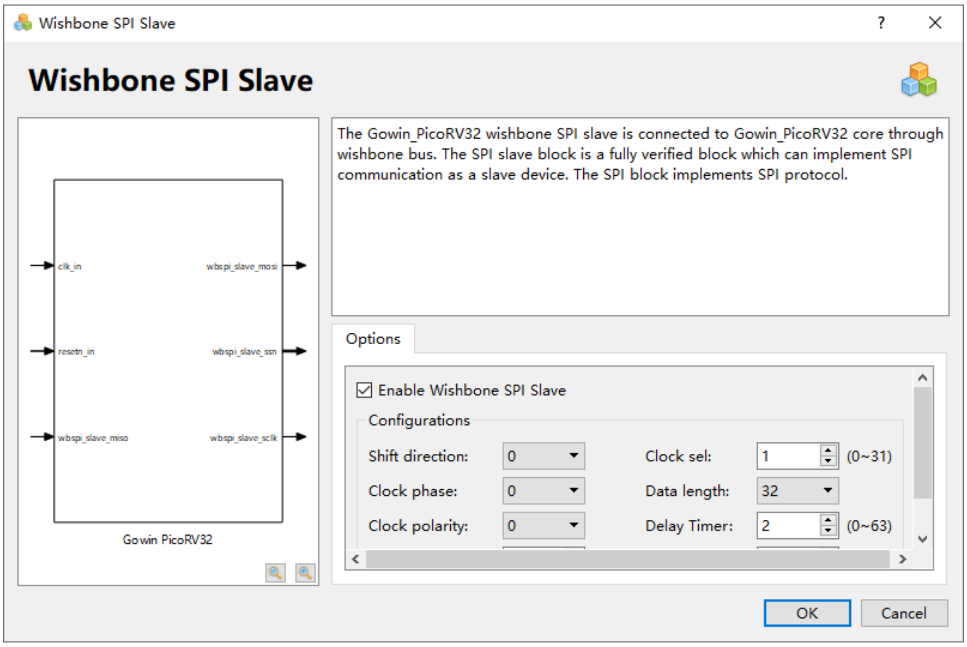
Parameter	Description
Shift direction	Specify data shift direction: When the value is 0, the Most Significant Bit (MSB) of data is shiftted first. When the value is 1, the Least Significant Bit (MSB) of data is shiftted first.
Clock phase	Specify clock phase of WB SPI Master: When the value is 0, the data is valid on the first edge of the SCLK When the value is 1, the data is valid on the second edge of the SCLK
Clock polarity	Specify clock polarity of WB SPI Master; When the value is 0, SCLK is low in the idle state When the value is 1, SCLK is high in the idle state
Clock count width	Specify clock count width There must be sufficient bit width to satisfy SCLK data width
Clock sel	Specify the frequency division factor required by SCLK produced by CLK_I frequency division SCLK frequency calculation: SCLK= CLK_I/(2*(CLOCK_SEL)+1) Value range: 0 ~ 2^(clock count width) - 1
Data length	Specify the bit width of the shift data Value range: 8/16/32/64

Parameter	Description
Slave number	Specify the supported Slave number Value range: 1 ~ 32
Delay time	Specify the delay time to wait before the first data transmission after the SS_N signal is valid: Delay time calculation: Delay = Delay Time * (SCLK period/2) Value range: 0 ~ 63
Interval length	Specify the number of SCLK cycles to wait for the SS_N signal after SPI transfers the request: Value range: 0 ~ 63

WB SPI Slave Configuration

Double-click WB SPI Slave to open the configuration view of Wishbone SPI Slave, as shown in Figure 3-15.

Figure 3-15 WB SPI Slave Configuration Option



The Gowin_PicoRV32 wishbone SPI slave is connected to Gowin_PicoRV32 core through wishbone bus. The SPI slave block is a fully verified block which can implement SPI communication as a slave device. The SPI block implements SPI protocol.

☑ Enable Wishbone SPI Slave

Configurations

Shift direction: 0 Clock sel: 1 (0~31)

Clock phase: 0 Data length: 32

Clock polarity: 0 Delay Timer: 2 (0~63)

- If "Enable Wishbone SPI Slave" option is selected, Gowin_PicoRV32 supports Wishbone SPI Slave. The default is disabled.
- If you select to enable "Wishbone SPI Slave", Wishbone SPI Slave parameters can be configured as shown in Table 3-4.

Table 3-4 WB SPI Slave Configuration Option

Parameter	Description
Shift direction	Specify data shift direction: When the value is 0, the Most Significant Bit (MSB) of data is shiftted first; When the value is 1, the Least Significant Bit (MSB) of data is shiftted first.
Clock phase	Specify clock phase of WB SPI Slave When the value is 0, the data is valid on the first edge of the

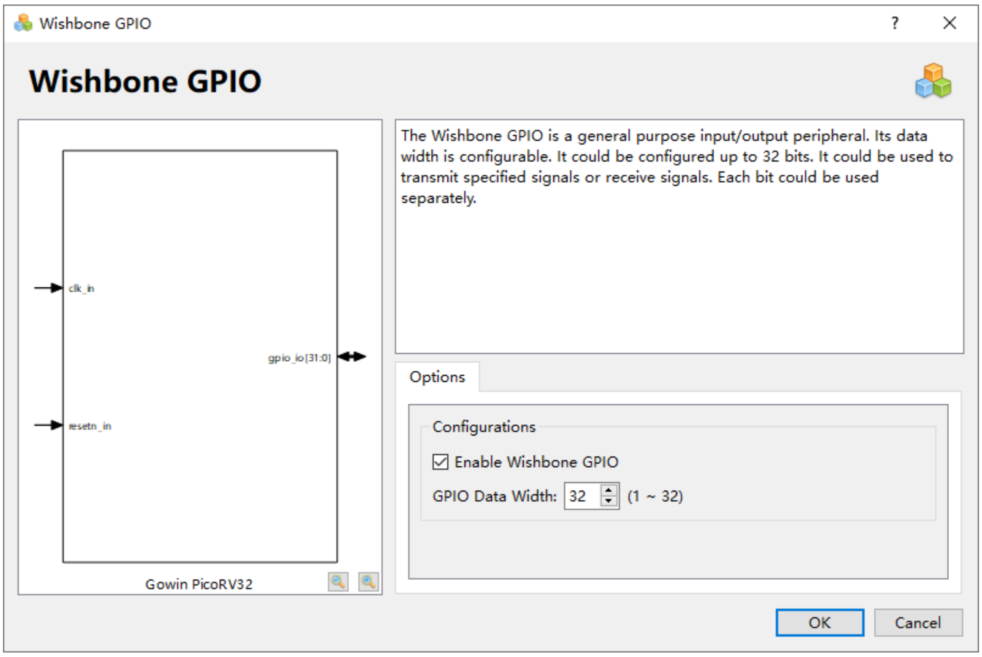
Parameter	Description
	SCLK; When the value is 1, the data is valid on the second edge of the SCLK.
Clock polarity	Specify clock polarity of WB SPI Slave: When the value is 0, SCLK is low in the idle state; .When the value is 1, SCLK is high in the idle state
Clock count width	Specify clock count width: There must be sufficient bit width to satisfy SCLK data width.
Clock sel	Specify the frequency division factor required by SCLK produced by CLK_I frequency division： SCLK frequency calculation: SCLK= CLK_I/(2*(CLOCK_SEL)+1) Value range: 0 ~ 2^(clock count width) - 1
Data length	Specify the bit width of the shift data: Value range: 8/16/32/64
Delay time	Specify the delay time to wait before the first data transmission after the SS_N signal is valid: Delay time calculation: Delay = Delay Time * (SCLK period/2) Value range: 0 ~ 63
Interval length	Specify the number of SCLK cycles to wait for the SS_N signal after SPI transfers the request. Value range: 0 ~ 63

WB GPIO Configuration

Double-click WB GPIO to open the configuration view of Wishbone GPIO, as shown in Figure 3-16.

- If "Enable Wishbone GPIO" option is selected, Gowin_PicoRV32 supports Wishbone GPIO. The default is disabled.
- If Wishbone GPIO is enabled, then configure GPIO data width of Wishbone GPIO ranging from 1 to 32.

Figure 3-16 WB GPIO Configuration Option



The Wishbone GPIO is a general purpose input/output peripheral. Its data width is configurable. It could be configured up to 32 bits. It could be used to transmit specified signals or receive signals. Each bit could be used separately.

Configurations

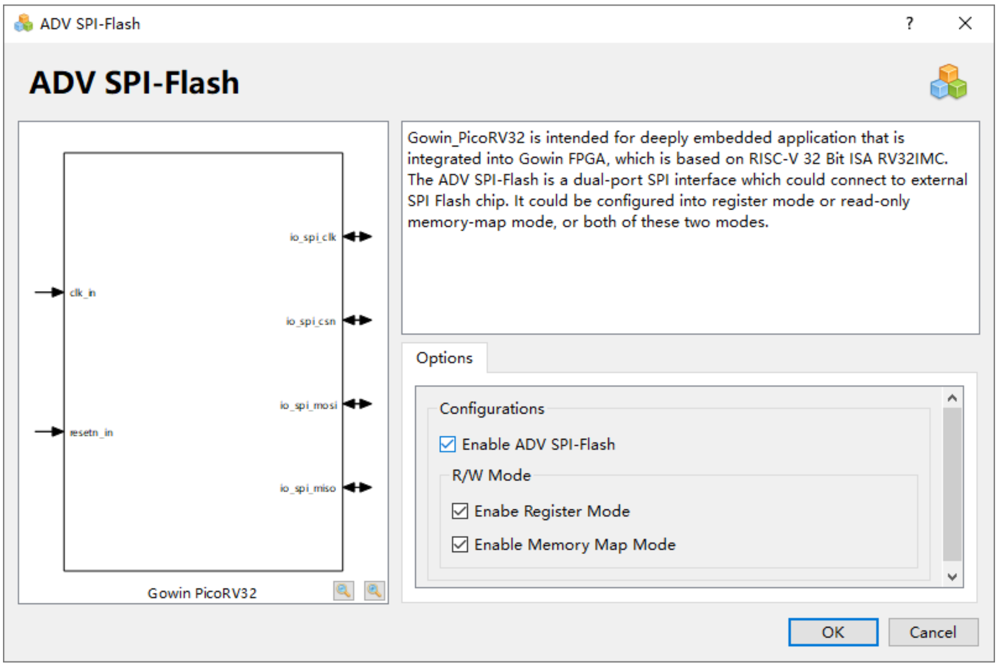
☑ Enable Wishbone GPIO

GPIO Data Width: 32 (1 ~ 32)

ADV SPI-Flash Configuration

Double-click ADV SPI-Flash to open the configuration view of ADV SPI-Flash Memory, as shown in Figure 3-17.

Figure 3-17 ADV SPI-Flash Configuration Option



Gowin_PicoRV32 is intended for deeply embedded application that is integrated into Gowin FPGA, which is based on RISC-V 32 Bit ISA RV32IMC. The ADV SPI-Flash is a dual-port SPI interface which could connect to external SPI Flash chip. It could be configured into register mode or read-only memory-map mode, or both of these two modes.

Configurations

☑ Enable ADV SPI-Flash

R/W Mode

☑ Enabe Register Mode

☑ Enable Memory Map Mode

- If "Enable ADV SPI-Flash" option is selected, Gowin_PicoRV32 supports ADV SPI-Flash Memory. The default is disabled.
- ADV SPI-Flash supports MCU software programming design download startup and run.
- ADV SPI-Flash supports read, write, and erasure Memory.

- The read and write configuration of Memory is shown in Table 3-5.

Table 3-5 ADV SPI-Flash R/W Configuration Option

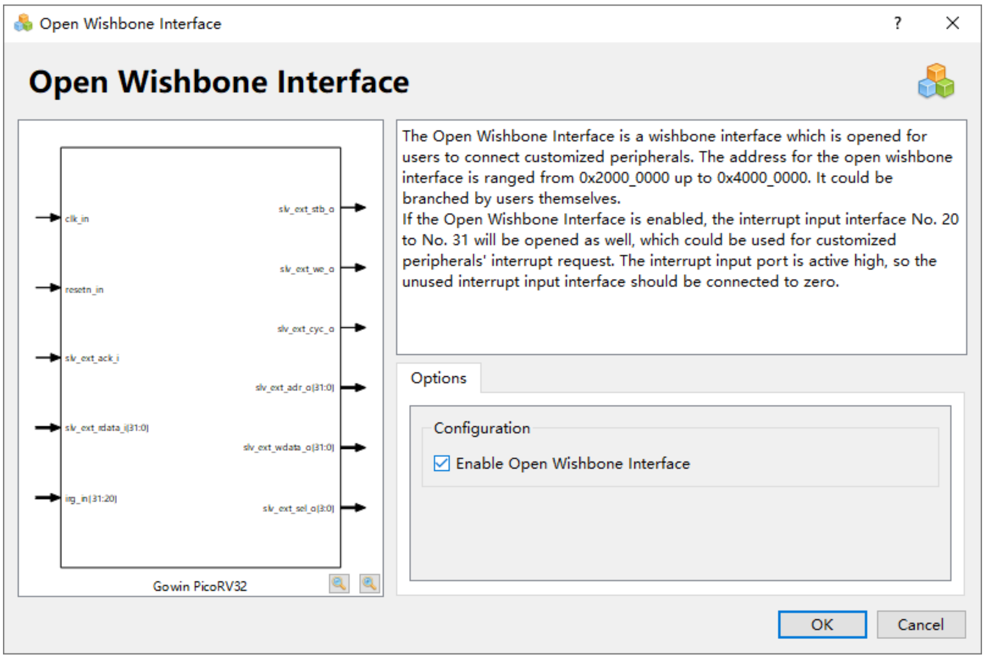
Parameter	Description
Enable Register Mode	Enables Register Mode, selected by default.
Enable Memory Map Mode	Enables Memory Map Mode, selected by default.

OPEN WB INTERFACE Configuration

Double-click to open OPEN WB INTERFACE. You can configure Open Wishbone Interface, as shown in Figure 3-18.

- The default is disabled.
- If "Enable Open Wishbone Interface" option is selected, Gowin_PicoRV32 supports "OPEN WB INTERFACE". You can connect peripherals used to extend Wishbone bus interface on this interface.
- There are 12 external interrupt signals irq_in[31:20] reserved for peripherals.

Figure 3-18 OPEN WB INTERFACE Configuration Option



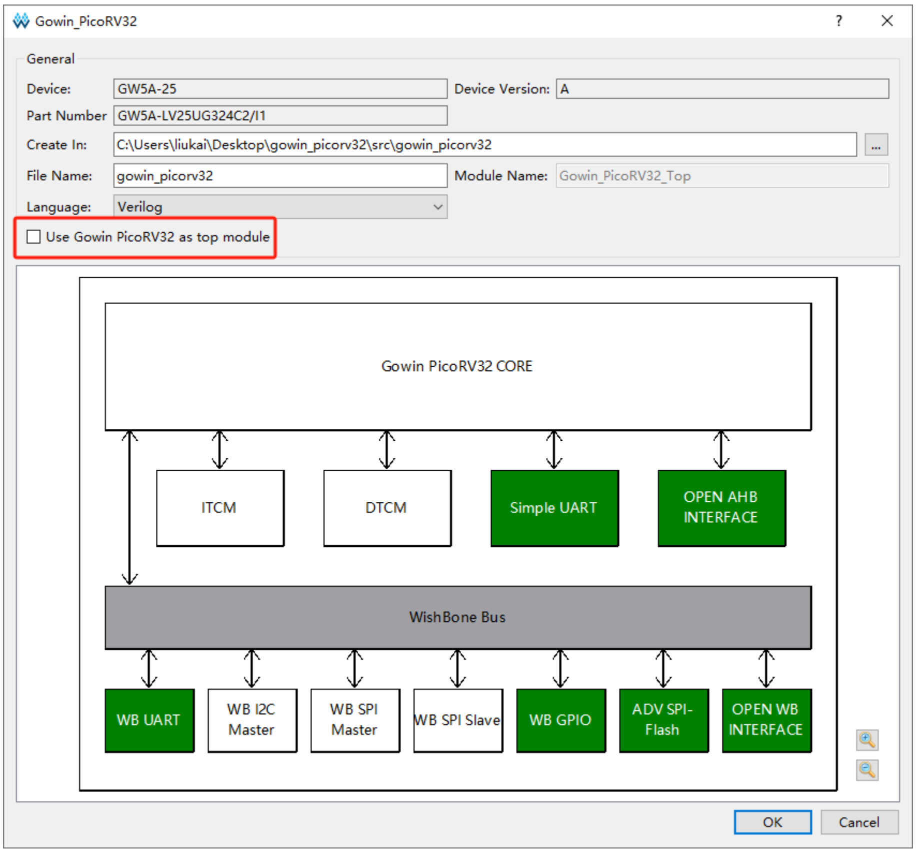
The Open Wishbone Interface is a wishbone interface which is opened for users to connect customized peripherals. The address for the open wishbone interface is ranged from 0x2000_0000 up to 0x4000_0000. It could be branched by users themselves.
If the Open Wishbone Interface is enabled, the interrupt input interface No. 20 to No. 31 will be opened as well, which could be used for customized peripherals' interrupt request. The interrupt input port is active high, so the unused interrupt input interface should be connected to zero.

☑ Enable Open Wishbone Interface

Top Module Configuration

- If Gowin_PicoRV32 is the top module, then select "Use Gowin PicoRV32 as Top Module" to set Gowin_PicoRV32 as Top Module.
- If you configure OPEN WB INTERFACE or OPEN AHB INTERFACE to extend the peripherals of Wishbone bus interface or AHB bus interface, disable "Use Gowin PicoRV32 as top module", as shown in Figure 3-19.

Figure 3-19 Top Module Configuration Option



Device: GW5A-25
Device Version: A
Part Number GW5A-LV25UG324C2/I1
Create In: C:\Users\liukai\Desktop\gowin_picorv32\src\gowin_picorv32
File Name: gowin_picorv32
Module Name: Gowin_PicoRV32_Top
Language: Verilog
☐ Use Gowin PicoRV32 as top module

3.3 User Design

After Gowin_PicoRV32 IP Core configuration, Gowin_PicoRV32 IP design can be generated.

Instantiate Gowin_PicoRV32 Top Module, set it as Top Module or connect user design.

Import user designs and connect it with Gowin_PicoRV32 Top Module to form a complete RTL design.

3.4 Constraint

After the user RTL design is completed, physical constraints can be generated according to the used development board and the IO.

Timing constraints file can be generated according to timing requirements.

For how to generate physical constraints, see *SUG935，Gowin Design Physical Constraints User Guide*, *SUG1018，Arora Ⅴ Design Physical Constraints User Guide*.

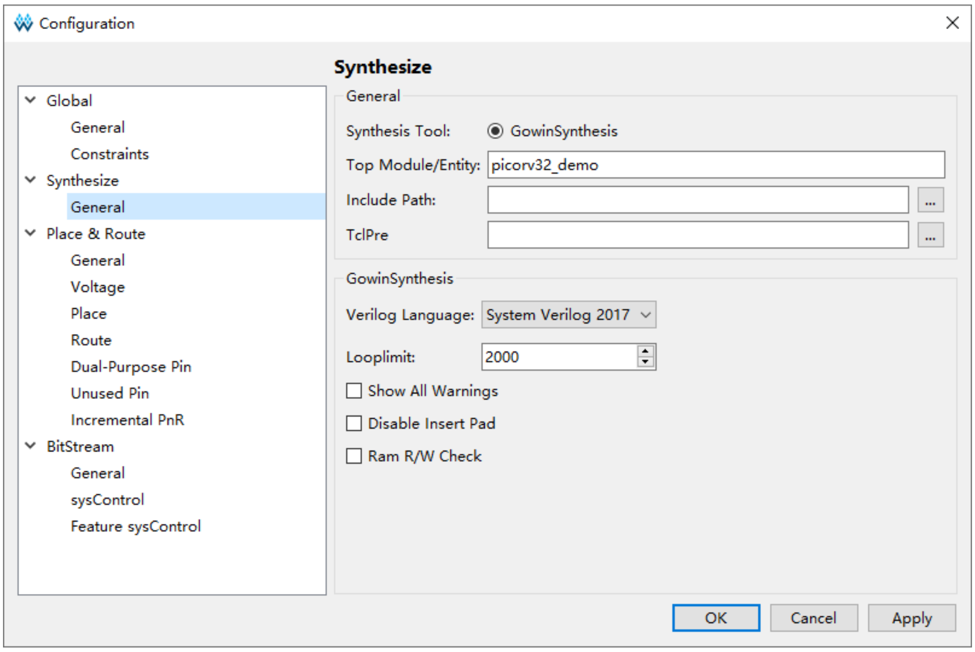
3.5 Project Configuration

3.5.1 Synthesis Configuration

The "Synthesize > General" configuration is as shown in Figure 3-20.

- Configure "Top Module/Entity" according to the actual top-level module name in the project.
- Configure "Include Path" according to the actual file path in the project.
- Configure "Verilog Language" according to System Verilog 2017.

Figure 3-20 Synthesis Configuration Option



Synthesize

General

Synthesis Tool: ● GowinSynthesis

Top Module/Entity: picorv32_demo

Include Path:

TclPre

GowinSynthesis

Verilog Language: System Verilog 2017

Looplimit: 2000

□ Show All Warnings

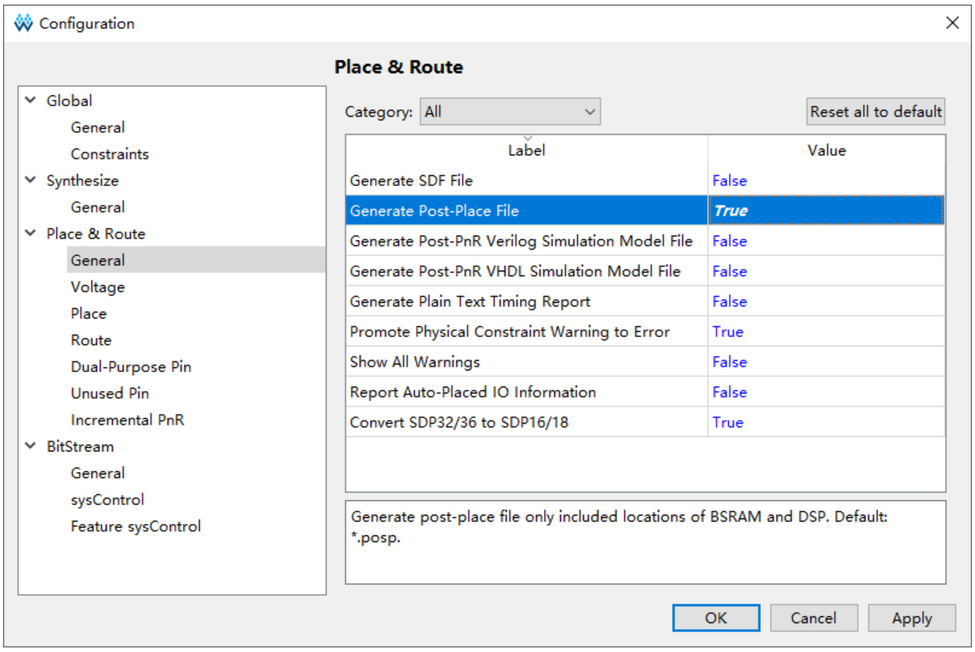
□ Disable Insert Pad

□ Ram R/W Check

3.5.2 Post-Place File Configuration

If you download the merged file of Gowin_PicoRV32 software design and hardware design automatically, configure "Place & Route > General > Generate Post-Place File" to "True" to generate Post-Place File, as shown in Figure 3-21.

Figure 3-21 Post-Place File Configuration Option



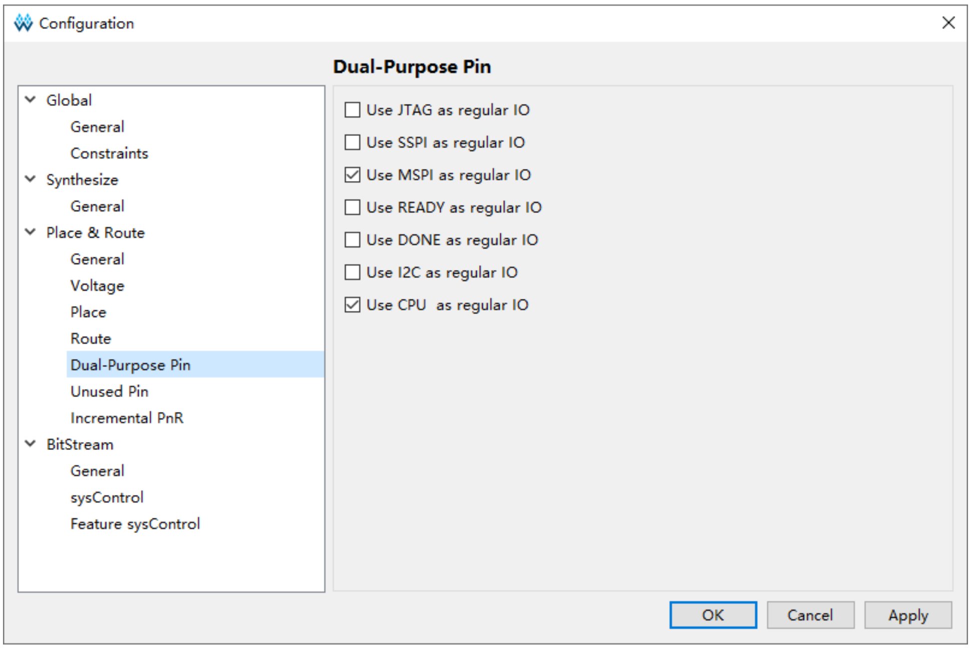
Label	Value
Generate SDF File	False
Generate Post-Place File	True
Generate Post-PnR Verilog Simulation Model File	False
Generate Post-PnR VHDL Simulation Model File	False
Generate Plain Text Timing Report	False
Promote Physical Constraint Warning to Error	True
Show All Warnings	False
Report Auto-Placed IO Information	False
Convert SDP32/36 to SDP16/18	True

Generate post-place file only included locations of BSRAM and DSP. Default: *.posp.

3.5.3 Dual-Purpose Pin Configuration

If Gowin_PicoRV32 is configured to "MCU boot from external Flash and run in ITCM" or "MCU boot and run in external Flash", you can configure dedicated IO as regular IO through "Place & Route > Dual-Purpose Pin" option, such as MSPI and CPU, as shown in Figure 3-22, or dual-purpose pin does not need to be configured.

Figure 3-22 Dual-Purpose Pin Configuration Option

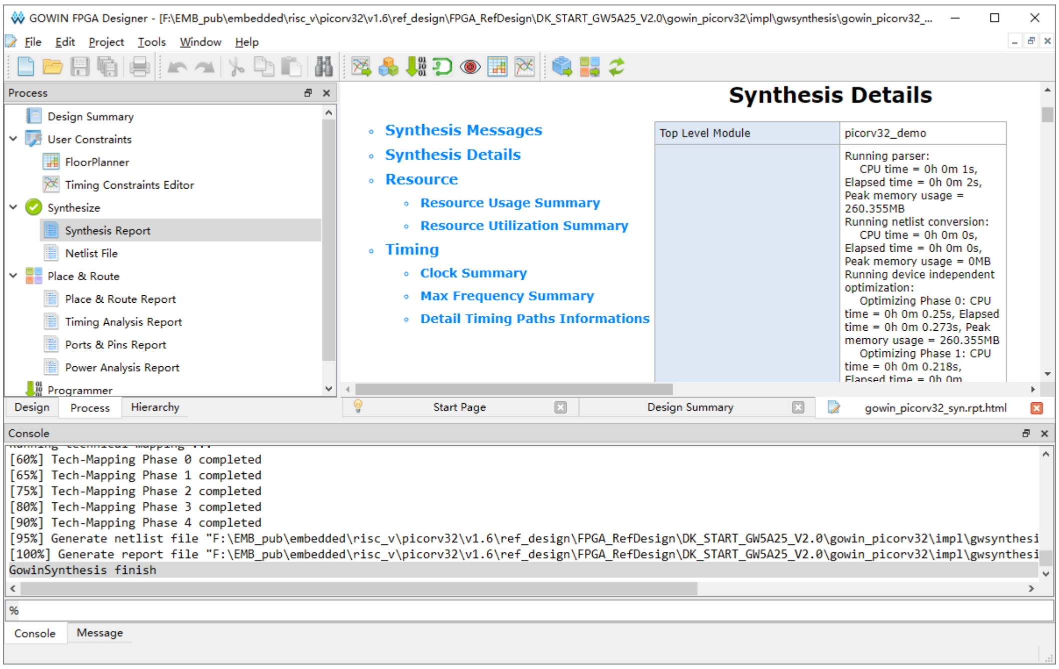


- □ Use JTAG as regular IO
- □ Use SSPI as regular IO
- ☑ Use MSPI as regular IO
- □ Use READY as regular IO
- □ Use DONE as regular IO
- □ Use I2C as regular IO
- ☑ Use CPU as regular IO

3.6 Synthesize

Run GowinSynthesis, the synthesis tool of Gowin Software, to complete the synthesis of RTL design and generate netlist files, as shown in Figure 3-23.

Figure 3-23 Synthesize



Top Level Module	picorv32_demo
	Running parser: CPU time = 0h 0m 1s, Elapsed time = 0h 0m 2s, Peak memory usage = 260.355MB Running netlist conversion: CPU time = 0h 0m 0s, Elapsed time = 0h 0m 0s, Peak memory usage = 0MB Running device independent optimization: Optimizing Phase 0: CPU time = 0h 0m 0.25s, Elapsed time = 0h 0m 0.273s, Peak memory usage = 260.355MB Optimizing Phase 1: CPU time = 0h 0m 0.218s, Elapsed time = 0h 0m

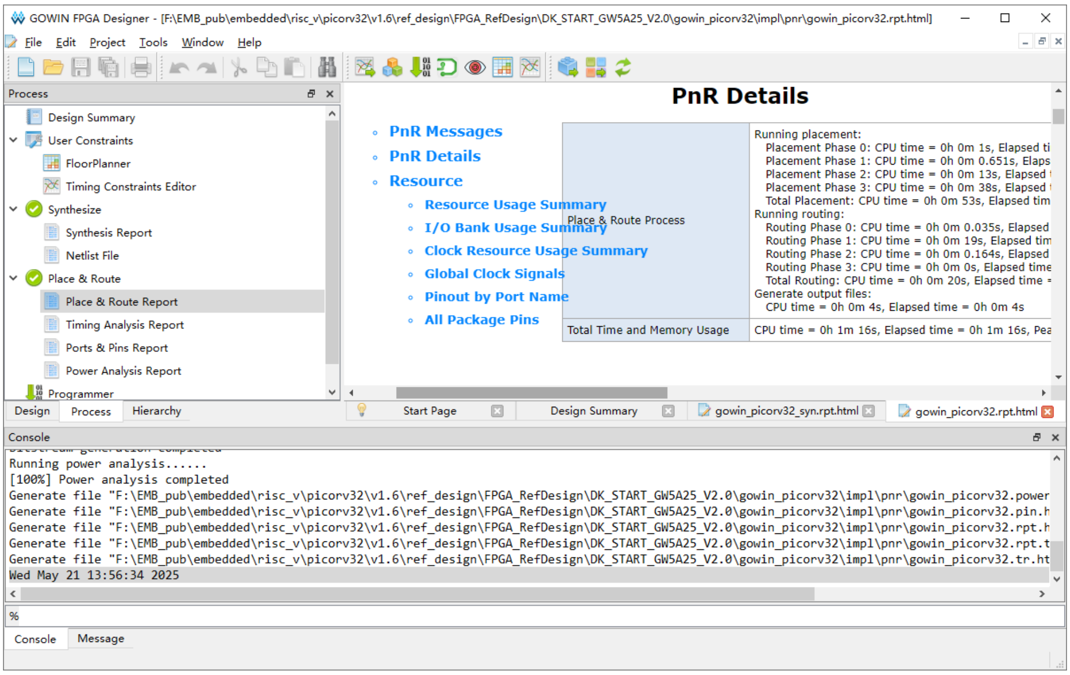
```
[60%] Tech-Mapping Phase 0 completed
[65%] Tech-Mapping Phase 1 completed
[75%] Tech-Mapping Phase 2 completed
[80%] Tech-Mapping Phase 3 completed
[90%] Tech-Mapping Phase 4 completed
[95%] Generate netlist file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\gwsynthesi
[100%] Generate report file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\gwsynthesi
GowinSynthesis finish
```

For the synthesis tool usage, please refer to *SUG100, Gowin Software User Guide*.

3.7 Place & Route

Run "Place & Route", the Place & Route tool of Gowin Software, to generate the bitstream files, as shown in Figure 3-24.

Figure 3-24 Place & Route



PnR Details

- PnR Messages
- PnR Details
- Resource
  - Resource Usage Summary
  - I/O Bank Usage Summary
  - Clock Resource Usage Summary
  - Global Clock Signals
  - Pinout by Port Name
  - All Package Pins

Place & Route Process	Running placement: Placement Phase 0: CPU time = 0h 0m 1s, Elapsed ti Placement Phase 1: CPU time = 0h 0m 0.651s, Elaps Placement Phase 2: CPU time = 0h 0m 13s, Elapsed Placement Phase 3: CPU time = 0h 0m 38s, Elapsed Total Placement: CPU time = 0h 0m 53s, Elapsed tim Running routing: Routing Phase 0: CPU time = 0h 0m 0.035s, Elapsed Routing Phase 1: CPU time = 0h 0m 19s, Elapsed tim Routing Phase 2: CPU time = 0h 0m 0.164s, Elapsed Routing Phase 3: CPU time = 0h 0m 0s, Elapsed time Total Routing: CPU time = 0h 0m 20s, Elapsed time = Generate output files: CPU time = 0h 0m 4s, Elapsed time = 0h 0m 4s
Total Time and Memory Usage	CPU time = 0h 1m 16s, Elapsed time = 0h 1m 16s, Pea

```
Running power analysis......
[100%] Power analysis completed
Generate file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\pnr\gowin_picorv32.power
Generate file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\pnr\gowin_picorv32.pin.h
Generate file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\pnr\gowin_picorv32.rpt.h
Generate file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\pnr\gowin_picorv32.rpt.t
Generate file "F:\EMB_pub\embedded\risc_v\picorv32\v1.6\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32\impl\pnr\gowin_picorv32.tr.ht
Wed May 21 13:56:34 2025
```

For the Place & Route tool usage, please refer to *SUG100, Gowin Software User Guide*.

3.8 Download

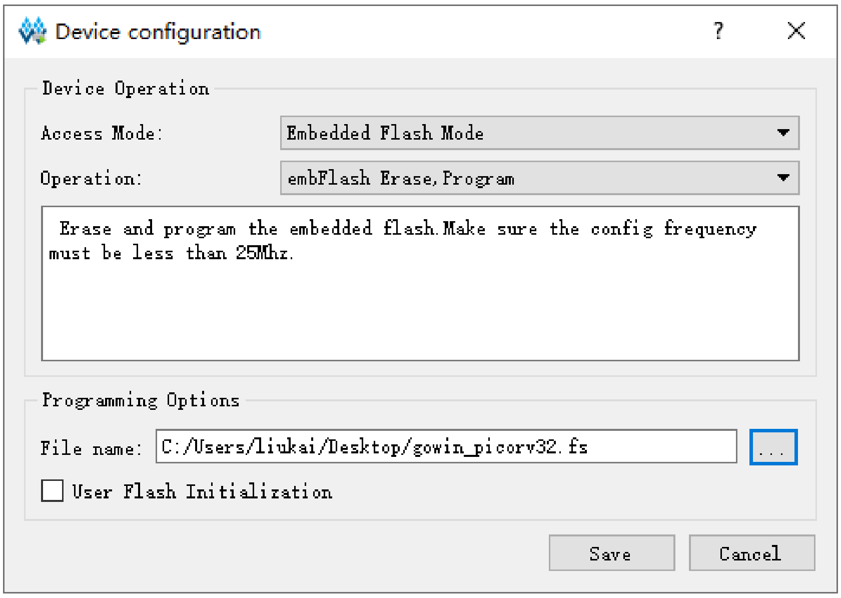
Run Programmer, the download tool of Gowin Software, to complete the download of bitstream files in hardware design.

Open programmer , and click "Edit > Configure Device" on the menu bar or "Configure Device" ( ) on the tool bar to open the "Device configuration".

Configuration options for LittleBee Family FPGA Products are as shown in Figure 3-25.

- Select "Embedded Flash Mode" from the "Access Mode" drop-down list.
- Select "embFlash Erase, Program" or "embFlash Erase, Program, Verify" from the "Operation" drop-down list.
- Import the hardware design bitstream file to be downloaded using "Programming Options > File name" option.
- Click "Save" to complete the configuration.

Figure 3-25 Configure Device for LittleBee Family FPGA Products



Access Mode: Embedded Flash Mode

Operation: embFlash Erase, Program

Erase and program the embedded flash.Make sure the config frequency must be less than 25Mhz.

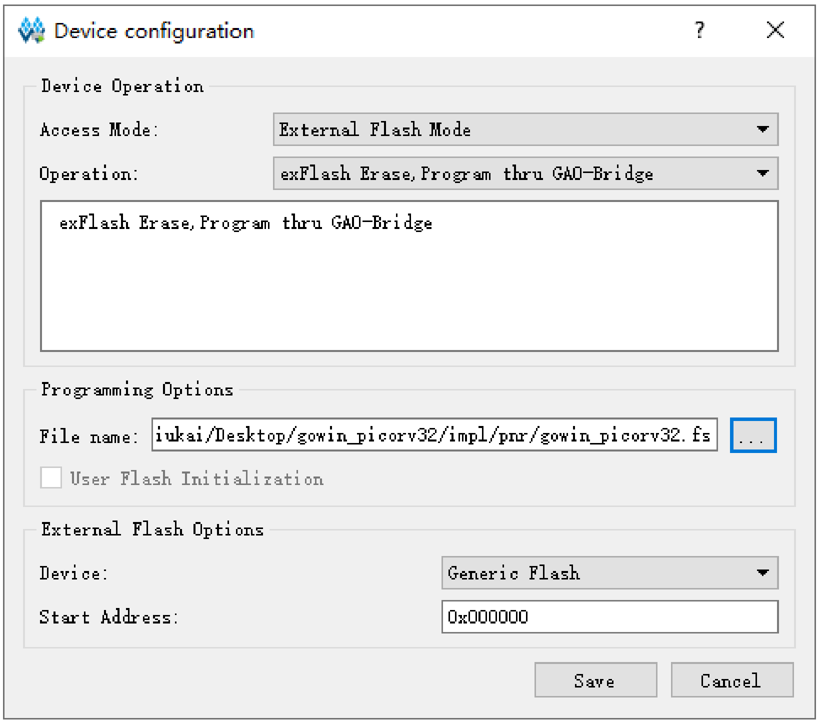
File name: C:/Users/liukai/Desktop/gowin_picorv32.fs

□ User Flash Initialization

Configuration options for Arora Family FPGA products are as shown in Figure 3-26.

- Select "External Flash Mode" from the "Access Mode" drop-down list.
- Select "exFlash Erase, Program thru GAO-Bridge" or "exFlash Erase, Program, Verify thru GAO-Bridge" from the "Operation" drop-down list.
- Import the hardware design bitstream file to be downloaded using "Programming Options > File name" option.
- Select "Generic Flash" from "External Flash Options > Device" option.
- Configure the start address as "0x000000" using "External Flash Options > Start Address" option.
- Click "Save" to complete the configuration.

Figure 3-26 Configure Device for Arora Family FPGA Products



Device configuration

Device Operation

Access Mode: External Flash Mode

Operation: exFlash Erase,Program thru GAO-Bridge

exFlash Erase,Program thru GAO-Bridge

Programming Options

File name: iukai/Desktop/gowin_picorv32/impl/pnr/gowin_picorv32.fs

☐ User Flash Initialization

External Flash Options

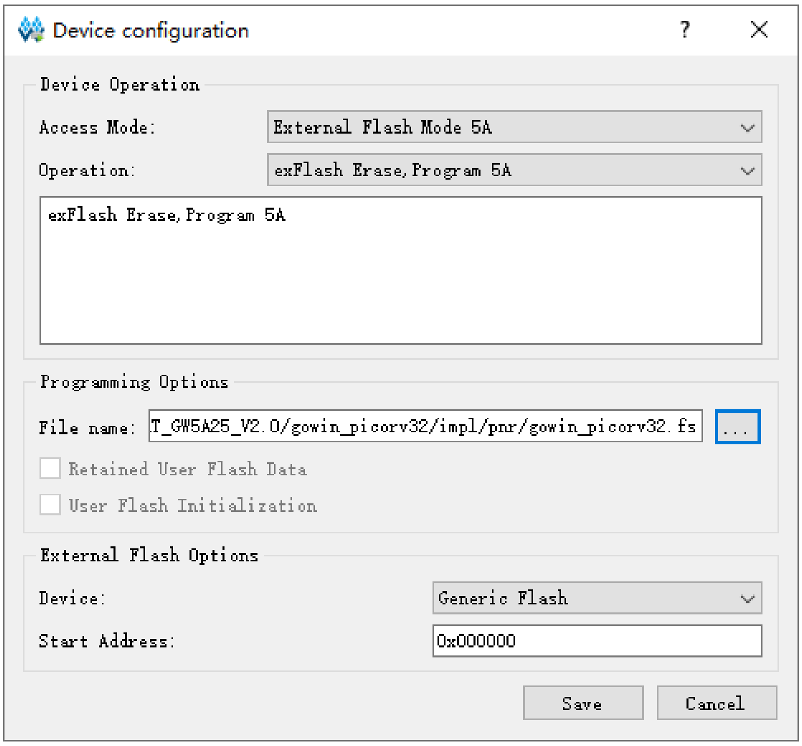
Device: Generic Flash

Start Address: 0x000000

Configuration options for Arora Ⅴ FPGA products are as shown in Figure 3-27.

- Select "External Flash Mode 5A" from the "Access Mode" drop-down list.
- Select "exFlash Erase, Program 5A" or "exFlash Erase, Program, Verify 5A" from the "Operation" drop-down list.
- Import the hardware design bitstream file to be downloaded using "Programming Options > File name" option.
- Select "Generic Flash" from "External Flash Options > Device" option.
- Configure the start address as "0x000000" using "External Flash Options > Start Address" option.
- Click "Save" to complete the configuration.

Figure 3-27 Configure Device for Arora V FPGA products



Device configuration

Device Operation

Access Mode: External Flash Mode 5A

Operation: exFlash Erase,Program 5A

exFlash Erase,Program 5A

Programming Options

File name: T_GW5A25_V2.0/gowin_picorv32/impl/pnr/gowin_picorv32.fs

□ Retained User Flash Data

□ User Flash Initialization

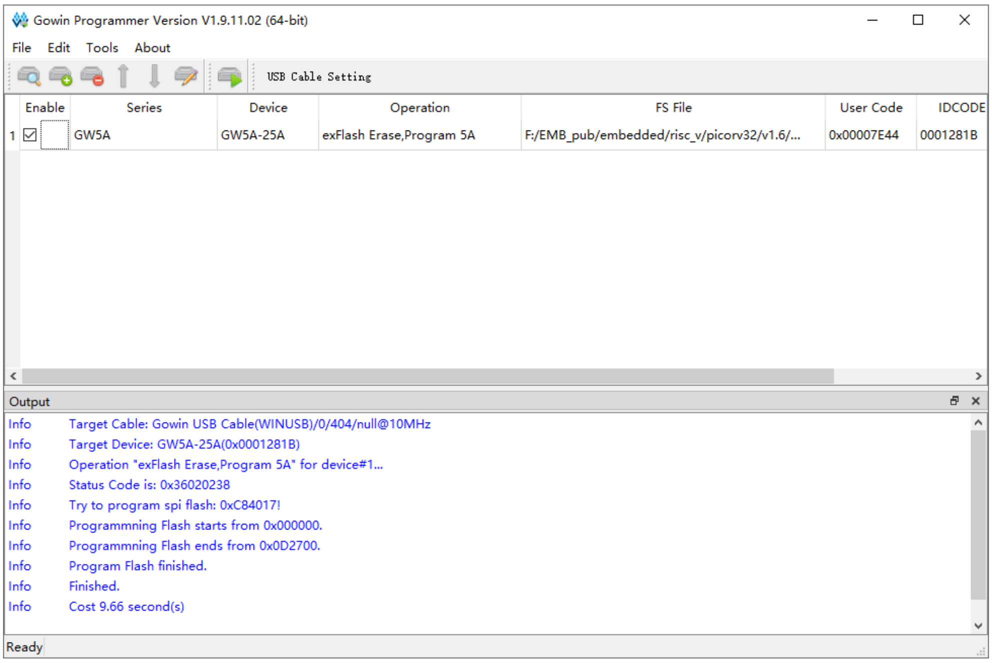
External Flash Options

Device: Generic Flash

Start Address: 0x000000

After device configuration, click "Program/Configure" ( ) in the Programmer toolbar to complete the download of bitstream files in hardware design, as shown in Figure 3-28.

Figure 3-28 Download



Gowin Programmer Version V1.9.11.02 (64-bit)

	Enable	Series	Device	Operation	FS File	User Code	IDCODE
1	☑	GW5A	GW5A-25A	exFlash Erase,Program 5A	F:/EMB_pub/embedded/risc_v/picorv32/v1.6/...	0x00007E44	0001281B

Output

```
Info      Target Cable: Gowin USB Cable(WINUSB)/0/404/null@10MHz
Info      Target Device: GW5A-25A(0x0001281B)
Info      Operation "exFlash Erase,Program 5A" for device#1...
Info      Status Code is: 0x36020238
Info      Try to program spi flash: 0xC84017!
Info      Programmning Flash starts from 0x000000.
Info      Programmning Flash ends from 0x0D2700.
Info      Program Flash finished.
Info      Finished.
Info      Cost 9.66 second(s)
```

Ready

For the usage of Programmer, see [SUG502, *Gowin Programmer User Guide*].

4 Reference Design

Gowin_PicoRV32 provides hardware reference design. Access the reference design through this link:

…\ref_design\FPGA_RefDesign\DK_START_GW2A18_V2.0\gowin_picorv32

…\ref_design\FPGA_RefDesign\DK_START_GW2A55_V1.3\gowin_picorv32

…\ref_design\FPGA_RefDesign\DK_START_GW5A25_V2.0\gowin_picorv32

…\ref_design\FPGA_RefDesign\DK_DP_GW5AT60_V1.0\gowin_picorv32

…\ref_design\FPGA_RefDesign\DK_START_GW5AST138_V1.0\gowin_picorv32

GOWIN

PROGRAMMING FOR THE FUTURE