Gowin FPGA Products

Programming and Configuration Guide

Revision History

Date	Version	Description
4/17/2017	1.00E	Initial version published.
5/31/2017	1.01E	• Configuration mode and value of different supported device updated; • RECONFIG N notes during programming built-in Flash updated.
10/13/2017	1.02E	Description of reusing pins updated.
3/16/2018	1.03E	GW1NS programming and configuration description added.
8/8/2018	1.04E	• The description of configuration process when Flash is empty updated; • Operation procedures for multiple configurations updated; • When MODE[0]=1, JTAG pins reuse description updated; • The programming features of B version devices updated; • Configuration notes and the timing for different configuration modes added.
1/8/2019	1.05E	• The configuration timing and parameters for SERIAL mode added; • The description of power supply requirements deleted.
8/16/2019	1.06E	• Power up description and configuration flow added; • The description of File Size Configuration modified.
5/15/2020	2.0E	• The note of JTAGSEL_N used as IO added. • GW1N(R)-2/GW1N(R)-2B/GW1N(R)-6 removed; • Configuration mode description optimized.
8/20/2020	2.1E	• JTAG Configuration added; • SSPI Configuration added; • AES Programming added;
10/30/2020	2.2E	Configuration File Loading Time added.
02/07/2021	2.3E	I^2C Configuration added.

Contents

List of Figures

List of Tables

1 About This Guide

1.1 Purpose

This guide mainly introduces general features and functions on programming and configuration of LittleBee® family devices and Arora family devices. It helps users to use Gowin FPGA products to their full potential.

1.2 Related Documents

The latest user guides are available on the GOWINSEMI Website. You can find the related documents at www.gowinsemi.com:

- DS100, GW1N series of FPGA Products Data Sheet
- DS102, GW2A series of FPGA Products Data Sheet
- DS117, GW1NR series of FPGA Products Data Sheet
- DS226, GW2AR series of FPGA Products Data Sheet
- DS961, GW2ANR series of FPGA Products Data Sheet
- DS821, GW1NS series of FPGA Products Data Sheet
- DS841, GW1NZ series of FPGA Products Data Sheet
- DS861, GW1NSR series of FPGA Products Data Sheet
- DS871, GW1NSE series of FPGA Products Data Sheet
- DS881, GW1NSER series of FPGA Products Data Sheet
- DS891, GW1NRF series of FPGA Products Data Sheet

1.3 Terminology and Abbreviations

The terminology and abbreviations used in this manual are as shown in Table 1-1.

Table 1-1 Abbreviations and Terminology

Terminology and Abbreviations	Full Name
LUT	Look-up Table
FPGA	Field Programmable Gate Array
JTAG	Joint Test Action Group
GPIO	General Purpose Input Output

Terminology and Abbreviations	Full Name
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
MSPI	Master Serial Peripheral Interface
SSPI	Slave Serial Peripheral Interface
CPU	Central Processing Unit
IEEE	Institute of Electrical and Electronics Engineers
ID	Identification
CRC	Cyclic Redundancy Check
FS file	Fuses file
Configuration	Configuration
Configuration Data	Configuration Data
Bitstream	Bitstream Data
Configuration Mode	Configuration Mode
EFlash/EmbFlash	Embedded Flash
Internal Flash	Internal Flash
Programming	Programming
Edit Mode	Edit Mode
User Mode	User Mode
Background Programming	Embedded Flash Background Programming
LSB	Least Significant Bit
MSB	Most Significant Bit
TAP	Test Access Port
Security Bit	Security Bit
Bscan	Boundary Scan
I2C (I^2C、IIC)	Inter-Integrated Circuits
SCL	Serial Clock
SDA	Serial Data

1.4 Support and Feedback

Gowin Semiconductor provides customers with comprehensive technical support. If you have any questions, comments, or suggestions, please feel free to contact us directly by the following ways.

Website: www.gowinsemi.com

E-mail:support@gowinsemi.com

2 Glossary

This chapter presents an overview of the terms that are commonly used in the process of programming and configuring of Gowin FPGA products to help users get familiar with the related concepts.

Table 2-1 Glossary

Glossary	Meaning
Program	Write bitstream data generated by Gowin software to FPGA on-chip Flash or off-chip SPI Flash that is connected to the FPGA.
Configure	Load bitstream data generated by Gowin software to the FPGA SRAM via external interfaces or on-chip Flash.
GowinCONFIG	In addition to the generic JTAG configuration mode, Gowin FPGA products support additional configurations, including AUTO BOOT configuration, DUAL BOOT configuration, MSPI configuration, SSPI configuration, SERIAL configuration, and CPU configuration. How many GowinCONFIG configuration modes each device supports depend on the device model and package.
MODE[2:0]	A representation of the three MODE pin values associated with GowinCONFIG.
AUTO BOOT Configuration	FPGA loads bitstream data into the SRAM from an embedded Flash. Only non-volatile devices support this mode.
DUAL BOOT Configuration	Two bitstream files are stored in embedded Flash and external Flash separately. Switch to the embedded Flash if the external Flash fails to configure. Only non-volatile devices support this mode.
MSPI Configuration	As a master, FPGA is configured by reading bitstream from the external Flash via the SPI interface automatically.
SSPI Configuration	As a slave, FPGA is configured by the external master writing bitstream via the SPI interface.
SERIAL Configuration	As a slave, FPGA is configured by the external master writing bitstream via the serial interface.
CPU Configuration	As a slave, FPGA is configured by the external master writing bitstream via the parallel interface (8-bit).

Glossary	Meaning
I^{2}C Configuration	As a slave, FPGA is configured by the external master writing bitstream via the I^{2}C interface.
MULTI BOOT Configuration	The derivative concept of MSPI, it refers to that FPGA reads bitstream data from different addresses of external Flash. The loading address of the latter bitstream data is written in previous bitstream data and the configuration is completed by triggering RECONFIG_N to switch the data stream file under the condition that the device power is on. FPGA products that support MSPI all support this mode.
Remote Upgrade	After FPGA starts to work, if an upgrade is required, first write bitstream to embedded or external Flash through remote operation, and then FPGA reads the external Flash by triggering RECONFIG_N or powering up again to complete the configuration.
Daisy Chain	FPGA devices are connected sequentially in a serial way. Devices can be configured from the head of the chain in sequence according to the connection order, and data can only be transmitted between adjacent devices.
User Mode	Hands over control to users when the FPGA configuration has been completed. Only in user mode, configuration pins can be reused as GPIOs (Gowin Programmable I/O).
Edit Mode	FPGA can be programmed and configured in this mode. All configuration pins cannot be reused as GPIOs. The output of all GPIOs is high-impedance state, except transparent transmission.
ID CODE	Identification for the Gowin FPGA device. Each series of devices has a different number.
USER CODE	Used to identify the FPGA device that used. The user code can be written to the FPGA device through Gowin programmer. Up to 32-bit can be supported.
Security Bit	A special design for the configuration data security of Gowin FPGA product. After you write the bitstream with security bit to the device, no one will be able to read back the data. Gowin software sets a security bit for the bitstream data of all FPGA products by default.
Encryption	The Arora family of FPGA products supports this feature. After the encrypted bitstream is written to FPGA, the device will match the pre-stored key automatically, and then decrypt and wake up the device after successful matching. The device cannot work if matching fails.

3 Configuration Modes

3.1 LittleBee® Family of FPGA Products

Besides the JTAG configuration mode that is commonly used in the industry, the LittleBee® Family of FPGA products also support GOWINSEMI's own configuration mode: GowinCONFIG. GowinCONFIG configuration modes that are available and supported for each device depend on the device model and package. All non-volatile devices support JTAG and AUTO BOOT modes. Up to six configuration modes are supported, as shown in Table 3-1.

Table 3-1 Configuration Modes

Configuration Modes		MODE[2:0][1]	Description
JTAG		XXX[2]	The LittleBee® Family of FPGA products are configured via JTAG interface by external Host.
GowinCONFIG	AUTO BOOT	000	FPGA reads data from embedded Flash for configuration
	I²C[6]	100	External Host configure FPGA products via the I²C interface.
	SSPI	001	External Host configure FPGA products of LittleBee® Family via SPI interface.
	MSPI	010	As Master, FPGA reads data from external Flash (or other devices) via the SPI interface[3.]
	DUAL BOOT[4]	110	FPGA reads data from external Flash first and if the external Flash configuration fails, it reads from the Internal Flash.
	SERIAL[5]	101	External Host configure FPGA products of LittleBee® Family via DIN interface.
	CPU[5]	111	External Host configure FPGA products of LittleBee® Family via DBUS interface.

Note!

- [1] The unbound mode pins are grounded by default;
- [2] The JTAG configuration mode is independent of MODE value;
- [3] The SPI interfaces of the SSPI and MSPI modes are independent of each other;
- [4] Currently GW1N(R)-4 / GW1N(R)-4B do not support DUAL BOOT;
- [5] The CPU configuration mode and SERIAL configuration mode share SCLK, WE_N and CLKHOLD_N. The data bus pins for the CPU configuration mode share pins with MSPI and SSPI configuration modes.
- [6] I²C is only supported in some devices.

Note!

For details about configuration pins, pin reuse, and pin functions and application, please refer to 4 Configuration Pin.

3.2 Arora Family of FPGA Products

Besides the JTAG configuration mode that is commonly used in the industry, the Arora Family of FPGA products also support GOWINSEMI's own configuration mode: GowinCONFIG. The GowinCONFIG configuration modes that are available and supported for each device depend on the device model and package. The Arora Family of FPGA Products support bitstream encryption and security bit setting, which provides safety for user designs. The Arora Family FPGA products support bitstream decompression; users can compress bitstream to save storage memory.

Table 3-2 lists the configuration modes that are supported by the Arora Family FPGA products.

Table 3-2 Configuration Modes

Configuration Modes		MODE[2:0][1]	Description
JTAG		XXX[2]	External Host configures Arora Family of FPGA products via JTAG interface.
GowinCONFIG	MSPI	000	As Master, FPGA reads data from external Flash (or other devices) via the SPI interface[3].
	SSPI	001	External Host configures Arora Family of FPGA products via SPI interface.
	SERIAL[4]	101	External Host configures Arora Family of FPGA products via DIN interface.
	CPU[4]	111	External Host configures Arora Family of FPGA products via DBUS interface.

Note!

- [1] The unbound mode pins are grounded by default;
- [2] The JTAG configuration mode is independent of MODE value;
- [3] The SPI interfaces of the SSPI and MSPI modes are independent of each other;
- [4] The CPU configuration mode and SERIAL configuration mode share SCLK, WE_N and CLKHOLD_N. The data bus pins for the CPU configuration mode share pins with MSPI and SSPI configuration modes.

Note!

For details about configuration pins, pin reuse, and pin functions and application, please refer to 4 Configuration Pin.

4Configuration Pin

Gowin FPGA products have various configuration modes, including general JTAG configuration, active configuration, passive configuration, serial configuration and parallel configuration, etc., which can meet the various peripheral requirements of different users. The programming and configuration pins can be used as configuration pins and also can be reused as GPIO. Users can configure the pins as required. Users also can configure them according to their configuration functions to meet specific requirements.

4.1 Configuration Pin List and Reuse Options

4.1.1 Configuration Pin List

Table 4-1 contains a list of all the configuration pins of Gowin FPGA products together with the details of the pins used in each configuration mode and the shared pins in chip packages.

Table 4-1 Configuration Pin List

Pin Name	I/O	JTAG	GowinCONFIG						
			AUTO BOOT	I^{2}C	SSPI	MSPI	DUAL BOOT	SERIAL	CPU
RECONFIG_N	I	√	√		√	√	√	√	√
JTAGSEL_N	I	√							
TDO	O	√							
TMS	I	√							
TCK	I	√							
TDI	I	√							
READY	I/O	√	√		√	√	√	√	√
DONE	I/O	√	√		√	√	√	√	√
MODE[2:0]	I		√		√	√	√	√	√
SCLK	I				√			√	√

Pin Name	I/O	JTAG	GowinCONFIG						
			AUTO BOOT	I²C	SSPI	MSPI	DUAL BOOT	SERIAL	CPU
CLKHOLD_N/DIN	I				√			√	√
WE_N/DOUT	O							√	√
MI /D7	I/O					√			√
MO /D6	I/O					√			√
MCS_N /D5	I/O					√			√
MCLK /D4	I/O					√			√
FASTRD_N /D3	I/O					√			√
SI /D2	I/O				√				√
SO /D1	I/O				√				√
SSPI_CS_N/D0	I/O				√				√
SCL	I			√					
SDA	I/O			√					

Note!

- For the configuration modes supported by different devices, please refer to 3Configuration Modes;
- Please refer to 5Configuration Mode Introduction for the definition of each pin in different configuration modes.

4.1.2 Configuration Pin Multiplexing

To maximize the utilization of I/O, Gowin FPGA products support setting the configuration pins as GPIO pins. Before any configuration operation is performed on all series of Gowin FPGA products after power up, all related configuration pins are used as configuration pins by default. After successful configuration, the device enters into user mode and reassigns the pin functions according to the multiplex options selected by the user.

Note!

When setting the pin multiplexing option, ensure the external initial connection state of the pins does not affect the device configuration. Isolate the connections that affect the configuration first, and then wait to modify them in user mode.

The reuse options for the configuration pins are detailed in Table 4-2.

Table 4-2 Pin Reuse Options

Name	Options	Description
JTAG PORT	Default Status	TMS, TCK, TDI, and TDO are used as dedicated configuration pins. JTAGSEL_N is used as GPIO.
	Set as GPIO	JTAGSEL_N pins are used as dedicated configuration pins: ● JTAGSEL_N=0, TMS, TCK, TDI, and TDO are used as configuration pins: ● JTAGSEL_N = 1, TMS, TCK, TDI, and TDO are used as GPIO after configuration.
I²C PORT	Default Status	SCL and SDA pins are used as dedicated configuration pins.
	Set as GPIO	SCL and SDA pins are used as GPIO after configuration.
SSPI PORT	Default Status	SCLK, CLKHOLD_N, SSPI_CS_N, SI and SO are used as dedicated configuration pins.
	Set as GPIO	SCLK, CLKHOLD_N, SSPI_CS_N, SI and SO are used as GPIO after configuration.
MSPI PORT	Default Status	FASTRD_N, MCLK, MCS_N, MO and MI are used as dedicated configuration pins.
	Set as GPIO	FASTRD_N, MCLK, MCS_N, MO and MI are used as GPIO after configuration.
RECONFIG_N	Default Status	Dedicated configuration pins.
	Set as GPIO	Used as GPIO after configuration.
READY	Default Status	Dedicated configuration pins.
	Set as GPIO	Used as GPIO after configuration.
DONE	Default Status	Dedicated configuration pins.
	Set as GPIO	Used as GPIO after configuration.

Note!

● [1] For the devices with JTAGSEL_N unbound, when debugging JTAG pin reuse, it's suggested to set MODE value to non-auto configuration mode (being neither auto-boot, dual boot, nor MSPI) before power up to avoid the other bit stream data affecting configuration. Device turns into user MODE, and JTAG pin changes into GPIO after power up and manually configuring JTAG. After the device is power up, the device enters User Mode, and the JTAG pin is used as GPIO. For the LittleBee® Family of FPGA products, when MODE[2: 0]=001, the JTAGSEL_N pin and the four JTAG Configuration pins (TCK, TMS, TDI, TDO) can be set as GPIOs simultaneously, but the JTAG pin cannot be recovered as a configuration pin by JTAGSEL_N. It can be recovered when the device reenters the edit mode.

● [2] The pins of SERIAL and CPU modes are shared with other configuration modes, so they cannot be set as GPIOs separately. However, the pins can be set as GPIOs in non-shared configuration modes.

Configure Dual-purpose Pin

The steps are as follows:

1. Open the project in Gowin software;
2. Select "Project > Configuration > Dual Purpose Pin" from the menu options, as shown in ;

3. Check the corresponding options.

Figure 4-1 Configuring Pin Reuse
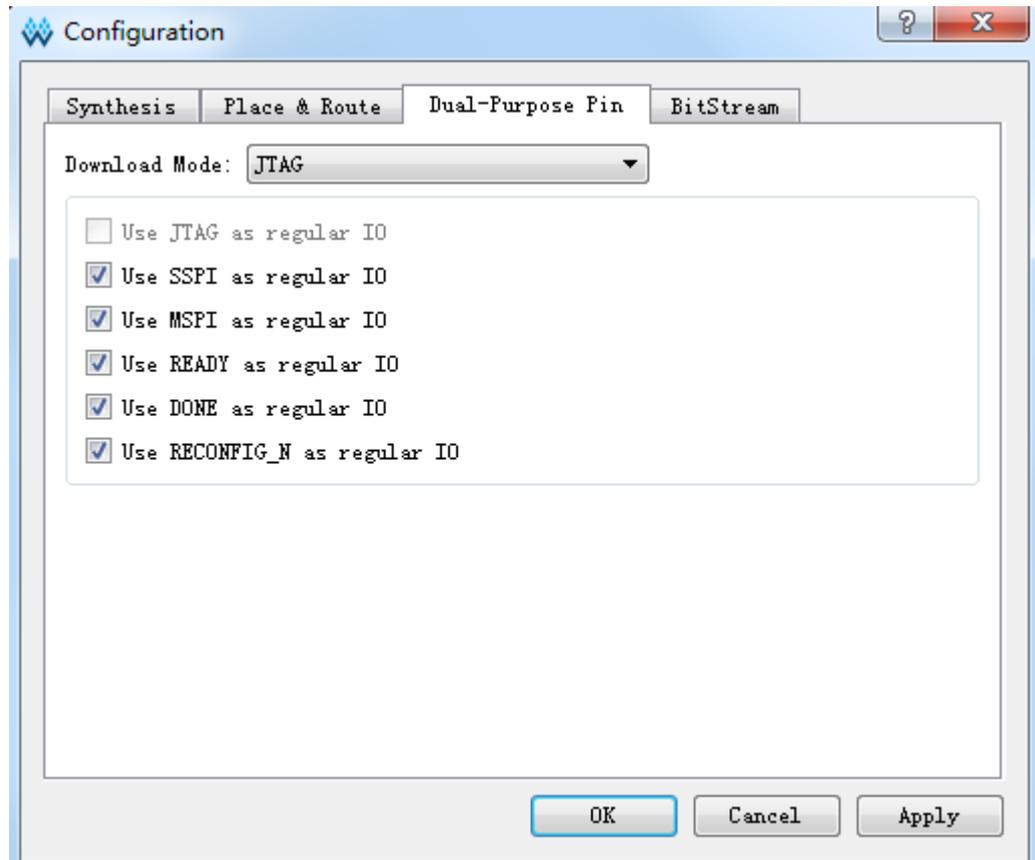
4.2 Configuration Pin Function and Application

The Pins RECONFIG_N, READY, and DONE pins are used in all configuration modes. Other pins can be set as dedicated pins or GPIO (Gowin Programmable IO) according to their specific application.

Table 4-3 Pin Function

Pin Name	Functional Description
RECONFIG_N	As a configuration pin, RECONFIG_N is an input pin that has an internal weak pull-up. Active low is used as the reset function for the FPGA programming configuration. **FPGA can't be configured if RECONFIG_N is set to low**. **Keep high-level during FPGA powering up until the powering up is stable for 1ms.** As a configuration pin, a low level signal with pulse width no less than 25ns is required for GowinCONFIG to reload bitstream data according to the MODE setting value. You can also write logic to control the pin to trigger the device to reconfigure as required. As a GPIO pin, RECONFIG_N can only be used as the output type. To ensure a smooth configuration, set the initial value of RECONFIG_N to high.
READY	In-out pins. Active-high. FPGA can be configured only when the READY signal is pulled up. When the READY signal is pulled down, recover the status by powering up or triggering RECONFIG_N.

Pin Name	Functional Description
	As an output configuration pin, it indicates that the FPGA can be configured or not. If the FPGA meets the configuration condition, the READY signal is high. If the configuration fails, READY signal is low. As an input configuration pin, you can delay the configuration via its own logic or by pulling down the READY signal. As a GPIO, it can be used as an input or output type. **If READY is used as an input GPIO, the initial value needs to be 1 before configuration. Otherwise, the FPGA cannot be configured.**
DONE	In-out pins. A signal which indicates FPGA is configured successfully, DONE is pulled up after successfully configuring. As an output configuration pin, it indicates the current configuration of FPGA: if configured successfully, the DONE signal is high and the device enters into working state. if the configuration fails, the DONE signal keeps low. As an input configuration pin, the user can delay the entering of user mode via its own internal logic or by reducing the DONE signal. When RECONFIG_N or READY signals are low, DONE signal also keeps low. When configuring SRAM using JTAG circuit, it does not need to take DONE signal into account. As a GPIO, it can be used as an input or output type. **If DONE is used as an input GPIO, the initial value of DONE should be 1 before configuring. Otherwise, the FPGA will fail to enter the user mode after being configured.**
MODE	GowinCONFIG modes selection pin. As the selection pin of GowinCONFIG modes, MODE is an input pin that has internal weak pull-up. The maximum bit width is 3 bits. When FPGA powers up or a low level pulse triggers RECONFIG_N, the device enters the corresponding GowinCONFIG mode in accordance with the MODE value. The same MODE value of the different Gowin series of FPGA products may have different configuration MODE. As the number of pins for each package is different, some MODE pins are not all bonded out, and the unbound MODE pins are grounded by default. Please refer to the corresponding PINOUT manual for further details. When MODE pins are used as GPIOs, they can be used as an input or output type. Note that when the MODE value changes, power-on again or providing one low pulse for triggering RECONFIG_N is required for it to take effect.
JTAGSEL_N	As a configuration pin, it is an input pin with internal weak pull-up. If JTAG pins are set as a GPIO in the Gowin software, the JTAG pins can become GPIOs after the device being powered up and successfully configured. The JTAG pin configuration functions can be recovered by pulling down JTAGSEL_N. The JTAG configuration functions are always available if no JTAG pin reuse is set. As a GPIO, it can be used as an input or output type. **Note!** The JTAGSEL_N pin and four JTAG pins (TCK, TMS, TDI, and TDO) are exclusive. JTAG pins can only be used as configuration pins if JTAGSEL_N is set as a GPIO. JTAGSEL_N can only be used as a configuration pin if JTAG pins are set as GPIOs.

Pin Name	Functional Description
	For the LittleBee® Family of FPGA products, when MODE[2: 0]=001, the JTAGSEL_N pin and the four JTAG pins (TCK, TMS, TDI, TDO) can be set as GPIOs simultaneously, but the JTAG pin cannot be recovered as a configuration pin by JTAGSEL_N. It can be recovered when the device reenters the edit mode.
TCK	As a configuration pin, it is an input pin. It is a serial clock input pin in the JTAG configuration mode. As a GPIO, it can be used as an input or output type.
TMS	As a configuration pin, it is an input pin with internal weak pull-up. It is a serial input pin in JTAG configuration mode. As a GPIO, it can be used as an input or output type.
TDI	As a configuration pin, it is an input pin with internal weak pull-up. It is a serial data input pin in JTAG configuration mode. As a GPIO, it can be used as an input or output type.
TDO	As a configuration pin, it is an output pin. It is a serial data output pin in JTAG configuration mode. As a GPIO, it can be used as an input or output type.
SCLK	As a configuration pin, it is an input pin. It is a clock input pin in SSPI, SERIAL, and CPU configuration modes. As a GPIO, it can be used as an input or output type.
CLKHOLD_N	As a configuration pin, it is an input pin with internal weak pull-up. It is a clock-locking pin in SSPI and CPU configuration modes: SCLK is valid when the input is high, and SCLK is invalid when the input is low. As a GPIO, it can be used as an input or output type.
SSPI_CS_N	As a configuration pin, it is an input pin with internal weak pull-up. It is a chip selection signal in the SSPI configuration mode, active low. As a GPIO, it can be used as an input or output type.
SI	As a configuration pin, it is an input pin. It is a serial data input pin in the SSPI configuration mode. As a GPIO, it can be used as an input or output type.
SO	As a configuration pin, it is an output pin. It is a serial data output pin in the SSPI configuration mode. As a GPIO, it can be used as an input or output type.

Pin Name	Functional Description
MCLK	As a configuration pin, it is an output pin. The output clock pin in the MSPI configuration mode is generated from a crystal oscillator in FPGA. The output frequency range of the crystal oscillator is 2.5 MHz ~ 125 MHz, and the default output frequency is 2.5 MHz. The MSPI configuration mode does not support 125 MHz clock. Please refer to the corresponding device datasheet for further detailed data on the on-chip crystal oscillator. The MCLK frequency values can be modified through the Gowin software interface, as shown in Figure 4-2. Open Gowin software, select "Project > Configuration" from the menu options, click "BitStream" and select the MCLK frequency values from the "Download Speed" pull-down list. As a GPIO, it can be used as an input or output type. **Figure 4-2 MCLK Frequency Setting**
MCS_N	As a configuration pin, it is an output pin. It is a chip selection signal in MSPI configuration mode, active low. As a GPIO, it can be used as an input or output type.
MI	As a configuration pin, it is an input pin. It is a serial data input pin in MSPI configuration mode. As a GPIO, it can be used as an input or output type.
MO	As a configuration pin, it is an output pin. Serial data output pin in MSPI configuration mode. As a GPIO, it can be used as an input or output type.
FASTRD_N	As a configuration pin, it is an input pin. In the MSPI mode, FASTRD_N is used to select Flash access speed. High indicates regular Flash access mode(command 0x03). Low indicates high-speed Flash access mode; The high-speed flash access command of each manufacturer is different. Please refer to the corresponding Flash manual. As a GPIO, it can be used as an input or output type.
WE_N	As a configuration pin, it is an input pin. Select the data input/output of D[7:0] in CPU mode: Read operation when WE_N is high; write operation when WE_N is low. As a GPIO, it can be used as an input or output type.
D0=D7	In-out pins.
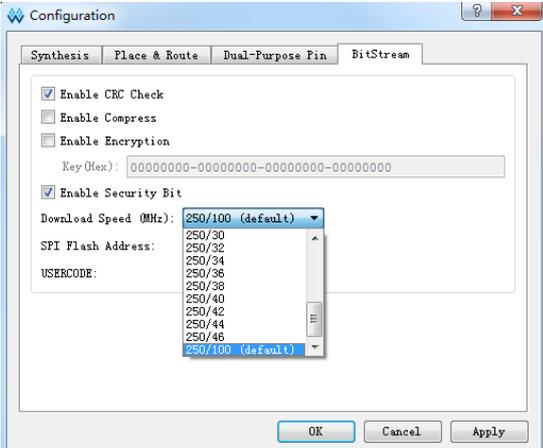
Pin Name	Functional Description
	Data input/output pins in CPU configuration mode, 8-bit width. Determine the input/output of D0 ~ D7 according to WE_N. As a GPIO, it can be used as an input or output type.
DIN	As a configuration pin, it is an input pin with internal weak pull-up. It is a serial data input pin in the SERIAL configuration mode. As a GPIO, it can be used as an input or output type.
DOUT	As a configuration pin, it is an output pin. It is a serial data output pin in the SERIAL configuration mode, which is only used as the input to the latter device when the FPGA is cascading. As a GPIO, it can be used as an input or output type.
SCL	As a configuration pin, it is an input pin. As a GPIO, it can be used as an input type.
SDA	As a configuration pin, it is an in/out pin. As a GPIO, it can be used as an input or output type.

5 Configuration Mode Introduction

Gowin FPGA products include the SRAM-based high-performance Arora Family of FPGA products and small capacity nonvolatile device of the LittleBee® Family of FPGA products with embedded Flash. Any configuration data that is stored in the SRAM device is lost after it is powered down; as such, it will need to be reconfigured each time it is powered up. The data stored on non-volatile devices with built-in flash will be stored in the chip if the device is powered down, and the device can be automatically reconfigured after power up via the AUTOBOOT or DUALBOOT configuration options.

Gowin FPGA products have abundant packages. The configuration modes supported by each device are related to the number of configuration pins bonded out: All devices support JTAG configuration, but only non-volatile devices support AUTO BOOT or DUAL BOOT configuration. The mode value for each configuration is different.

5.1 Configuration Notes

GOWINSEMI FPGA products include LittleBee® family and Arora family. Whether the name of the device contains R does not affect the configuration feature, the main difference is that SDRAM/PSRAM is integrated in all FPGA products that have a serial number that includes the letter R. Except DUALBOOT configuration features, the GW1NS series of FPGA products have same features as the GW1N series.

Power Up and Configuration Flow

When the power up voltage of VCC, VCCIO, and VCCX reaches the min. value, FPGA begins to start: stable voltage and RECONFIG_N is not pulled down > The internal circuit of FPGA pulls down READY and DONE pins > FPGA initialization > Pulling up READY and sampling MODE value > Reading and checking the configuration data according to the configuration mode > FPGA waking up > DONE pulling up > Entering user mode.

Power supply voltage needs to be stable in the process of FPGA start-up. RECONFIG_N needs to keep high after being powered up until the voltage is stable for 1ms and also in the process of FPGA initialization.

RECONFIG_N can be left floating or externally pulled up. All GPIOs output high impedance state before FPGA is waken up.

GOWINSEMI FPGA products write bitstream data to SRAM, on-chip Flash, or off-chip Flash according to the data storage and the instructions. Only the LittleBee® Family of FPGA products support operations on on-chip Flash. All products support operations on SRAM and external Flash.

SRAM Operation

The SRAM operations include read device ID CODE and USER CODE, read device status register information and SRAM configuration. The device ID needs to be verified before configuration. Only the device with successful ID verification can be configured. The USER CODE is the identification number for users to distinguish between the devices that share the same ID CODE. The state register of the device records the status information before and after FPGA configuration, and you can use this information to analyze the state of the device accordingly. Please refer to Table 5-10 for the meaning of the status register. During SRAM configuration, only the bitstream data with no security bit setting supports validation. Data with security bit cannot be readback or verified.

On-chip/Off-chip Flash Operation

The built-in flash operations include erasing, programming and verification. The built-in flash can only be programmed via the JTAG interface, and the clock rate is no less than 1MHz. Please refer to Table 5-7 for the clock rate.

Note!

During configuring SRAM devices via built-in Flash (AUTOBOOT configuration and DUALBOOT configuration) and programming built-in Flash, the FPGA needs to remain powered up, and the RECONFIG_N cannot be triggered at low level; otherwise, it may cause irreparable damages to the built-in Flash.

It is required to clear the SRAM content before programming the embedded Flash or external Flash of the A version of LittleBee® family devices. The B version of LittleBee® family devices supports the feature of transparent transmission. That is to say, the B version device can program the embedded Flash or external Flash via the JTAG interface without affecting the current working state. During programming, the B version device works according to the previous configuration. After programming, RECONFIG_N is triggered at low pulse to complete the online upgrade. This feature applies to the applications requiring long online time and irregular upgrades.

Dual-purpose Pin Configuration

In different configuration modes, users need to ensure that FPGA works in the selected configuration mode according to the pin functions. If user pins is insufficient, these pins can be configured and used as GPIOs, but pins associated with data transmission need to be kept. MODE [2:0] is used to select the GowinCONFIG programming configuration MODE. MODE can be fixed through pull-up or pull-down resister. It is recommended to use 4.7 K pull-up resister and1 K pull-down resistor.

Note!

The RECONFIG_N, READY, and DONE pins are associated with each configuration mode. Whether they are set as GPIO or not, users should ensure that their initial value or pin connection state meets programming and configuration conditions before completing the configuration process.

Recommended Pin Connection

When users are designing a circuit schematic diagram, the recommended connection is as shown in Figure 5-1.

Figure 5-1 Recommended Pin Connection
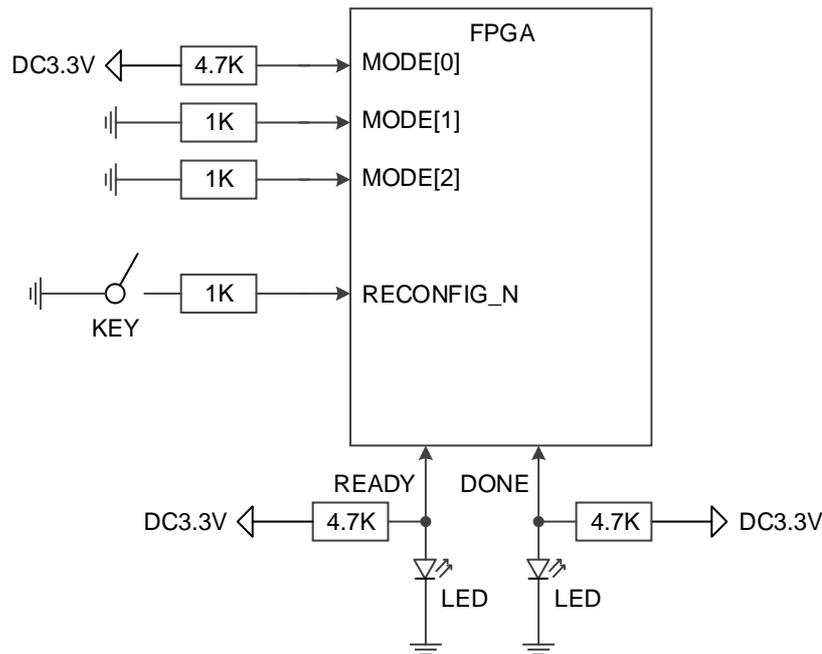
Note!

- Add the dial switch to change the MODE value; Some MODE pins of devices are not all bonded out, and the unbonded MODE pins are grounded by default;
- The values of READY and DONE signals have no meaningful reference in JTAG configuration.
- The unbonded RECONFIG_N, READY, and DONE pins have been internally processed, with no influence on the configuration function.

Timing for Power-on Again and Triggering RECONFIG_N at Low Pulse

Figure 5-2 and Figure 5-3 show the timing for power-on again or triggering RECONFIG_N at low pulse.

Figure 5-2 Power Recycle Timing
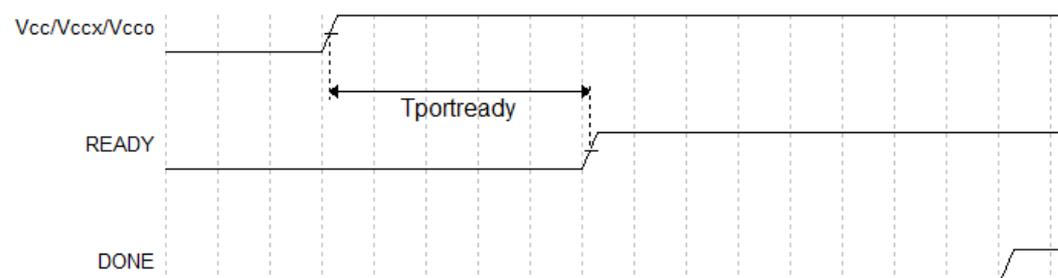
Figure 5-3 Trigger Timing
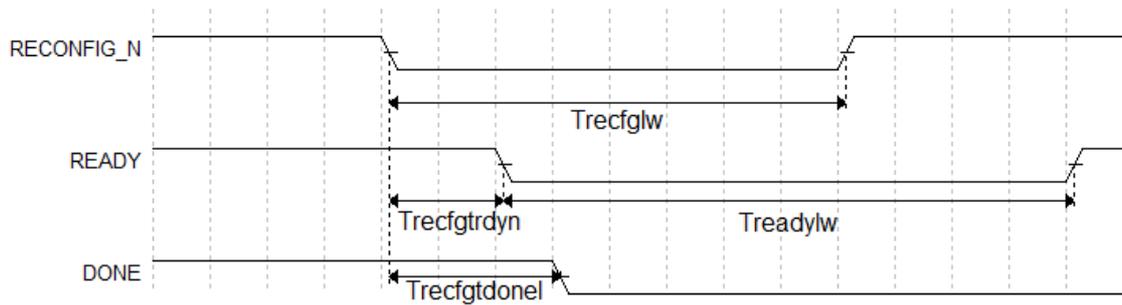
Timing parameters of the LittleBee® Family of FPGA Products is as shown in Table 5-1 .

Table 5-1 Timing Parameters for Cycling Power and RECONFIG_N Trigger

Name	Description	Min.	Max.
$T_{portready}$[1]	Time from application of V_{CC}, V_{CCX} and V_{CCO} to the rising edge of READY	50μs	200μs
$T_{recfglw}$	RECONFIG_N low pulse width	25ns	-
$T_{recfgtrdyn}$	Time from RECONFIG_N falling edge to READY low	-	70ns
$T_{readylw}$	READY low pulse width	TBD	-
$T_{recfgtdonel}$	Time from RECONFIG_N falling edge to READY low	-	80ns

Note!

In the case of MODE0=0, the device power-up waiting time is 200 μs; If MODE0=1, the device power-up waiting time is 50 μs.

Timing parameters of the Arora Family of FPGA Products are as shown in Table 5-2.

Table 5-2 Timing Parameters for Power-on again and RECONFIG_N Triggering (Arora Family)

Name	Description	Min.	Max.
$T_{portready}$	Time from application of V_{CC}, V_{CCX} and V_{CCO} to the rising edge of READY	-	23ms
$T_{recfglw}$	RECONFIG_N low pulse width	25ns	-
$T_{recfgtrdyn}$	Time from RECONFIG_N falling edge to READY low	-	70ns
$T_{readylw}$	READY low pulse width	TBD	-
$T_{recfgtdonel}$	Time from RECONFIG_N falling edge to READY low	-	80ns

5.2 JTAG Configuration

The JTAG configuration mode of Gowin FPGA products conforms to the IEEE1532 standard and the IEEE1149.1 boundary scan standard.

The JTAG configuration mode writes bitstream data to the SRAM of Gowin FPGA products. All configuration data is lost after the device is powered down. All Gowin FPGA products support the JTAG configuration

mode.

5.2.1 JTAG Configuration Mode Pins

The relevant pins for the JTAG configuration mode are shown in Table 5-3.

Table 5-3 Pin Description in JTAG Configuration Mode

Pin Name	I/O	Description
JTAGSEL_N[1]	I, internal weak pull-up	Revert JTAG pin from GPIO to configuration pin. Low active
TCK 2	I	JTAG serial clock input
TMS[2]	I, internal weak pull-up	JTAG serial mode input
TDI	I, internal weak pull-up	JTAG serial data input
TDO	O	JTAG serial data output

Note!

- [1] The JTAGSEL_N works only when the JTAG pin is set as a GPIO and the device starts to work. For the LittleBee® Family of FPGA products, when MODE[2：0]= 001, the JTAGSEL_N pin and the four JTAG pins (TCK, TMS, TDI, TDO) can be set as GPIOs simultaneously, but the JTAG pin cannot be recovered as a configuration pin by JTAGSEL_N. It can be recovered when the device reenters the editing mode.
- [2] TCK needs to connect 4.7 K pull down resister on the PCB.

5.2.2 Connection Diagram for the JTAG Configuration Mode

The connection diagram in the JTAG configuration mode is shown in Figure 5-4.

Figure 5-4 Connection Diagram for JTAG Configuration Mode
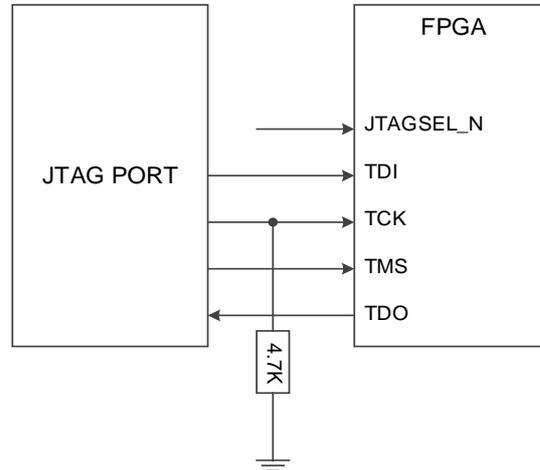
Note!

- If JTAGSEL_N is not bonded out, when debugging the JTAG pin reuse, it is suggested to set the MODE value to non-auto configuration mode (AUTOBOOT, DUALBOOT or MSPI) before powering up the device to avoid the other bitstream data affecting configuration. After power up and JTAG is configured manually, the device enters User MODE, and JTAG pin will be used as a GPIO.
- The clock frequency for JTAG configuration mode is no higher than 40MHz.

In addition to using JTAG to configure SRAM, the built-in Flash of Gowin non-volatile FPGA devices (LittleBee® Family) and the external SPI Flash of all other FPGA series programming can also be configured through the JTAG pin. The connection for programming the built-in Flash of the non-volatile devices is the same as that of the JTAG mode. Please refer to Figure 5-41 and 8 Boundary Scan for external SPI Flash programming.

In addition, Gowin FPGA products support JTAG daisy chain operation, which connects the FPGA TDO pin to the next FPGA TDI pin. Gowin programming software will identify the connected FPGA devices automatically and configure them in turn. The connection diagram for the daisy chain configuration is shown in Figure 5-5.

Figure 5-5 Connection Diagram of JTAG Daisy-Chain Configuration Mode
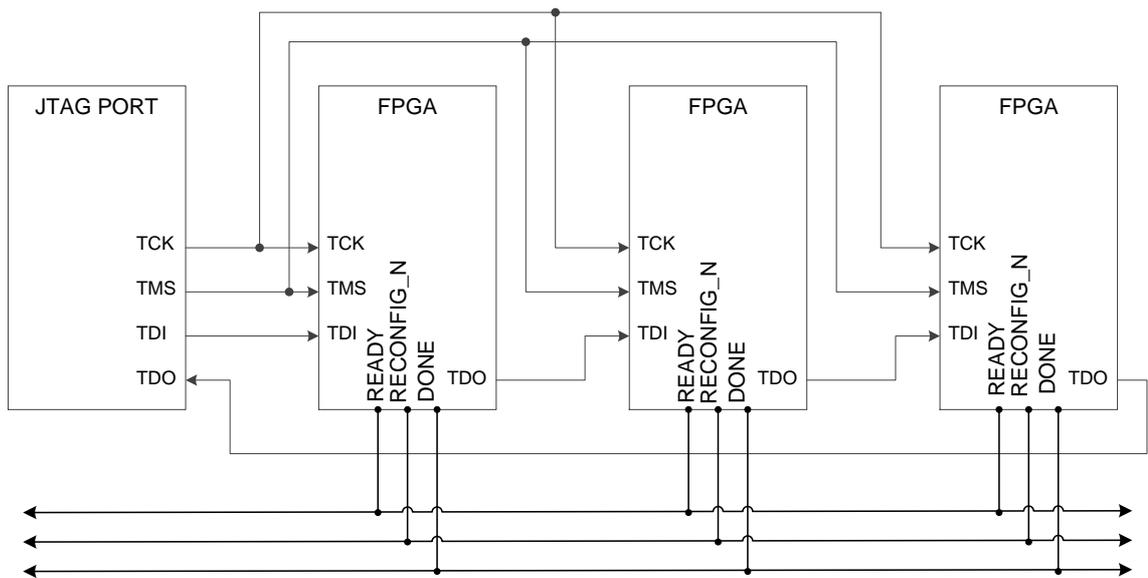
Note!

DONE, RECONFIG_N, and READY can be connected or not as appropriate.

5.2.3 JTAG Configuration Timing

See Figure 5-6 for the timing of JTAG mode.

Figure 5-6 JTAG Configuration timing
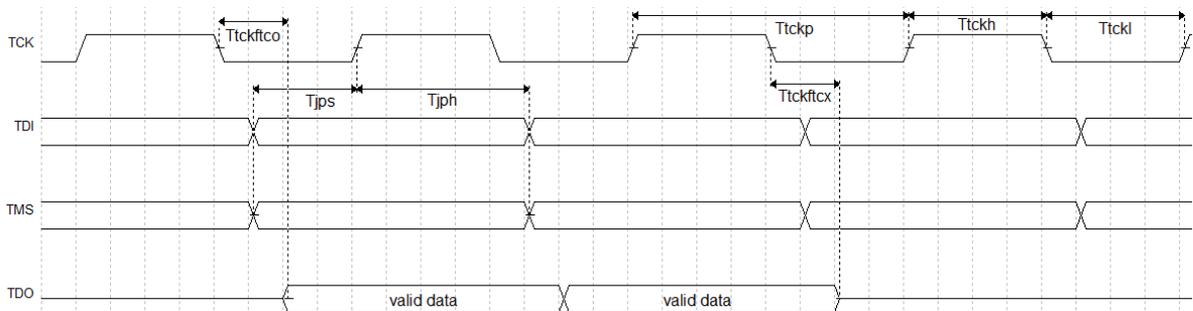
See Table 5-4 for the description of timing parameters.

Table 5-4 JTAG Configuration Timing Parameters

Name	Description	Min.	Max.
$T_{tckftco}$	Time from TCK falling edge to output	-	10ns
$T_{tckftcx}$	Time from SCLK falling edge to high impedance	-	10ns
T_{tckp}	TCK clock period	40ns	-
T_{tckh}	TCK clock high time	20ns	-
T_{tckl}	TCK clock low time	20ns	-
T_{jps}	JTAG PORT setup time	10ns	-
T_{jph}	JTAG PORT hold time	8ns	-

5.2.4 JTAG Configuration Process

TAP State Machine

The state machine for the test access port is designed to select an instruction register or a data register to connect it between TDI and TDO. In general, the instruction register is used to select the data register to be scanned. In the state machine diagram, the number on the side of the arrow indicates the logic state of the TMS when the TCK goes high, as shown in Figure 5-7.

Figure 5-7 TAP State Machine
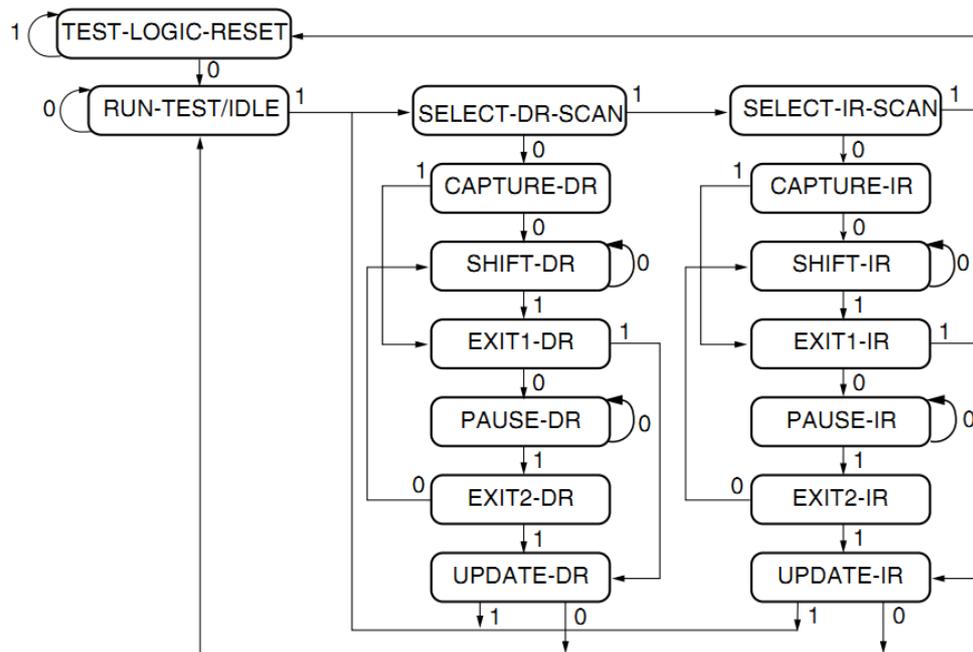
TAP Reset

After TMS keeps high (logic "1") and at least 5 strobes are input (higher and then low) at the TCK terminal, the TAP logic is reset, the TAP state machine in other states is converted into the state of test logic reset, and the JTAG port and the test logic are reset.

Note!

The CPU and peripherals are not reset in this state.

Note!

- The data on the TDO is valid from the falling edge of TCK in the Shift_DR or Shift_IR state;
- The data is not shifted in the Shift_DR or Shift_IR state;
- The data is shifted when leaving the Shift_DR or Shift_IR;
- The first to be shifted is the least significant bit (LSB) of the data;
- Once reset, all instructions will be reset or disabled.

Instruction Register and Data register

In addition to the test logic reset, the state machine can also control two basic operations:

● Instruction register (IR) scan;
● Data Register (DR) scan.

During the IR scanning operation, in Shift_IR state, the data or instructions are sent to the IR in the LSB way. The lower data bits are sent first. The instructions will be all sent when the sate machine returns to Run-Test-Idle, as shown in Figure 5-8

During the data register scanning operation, the data or instructions are sent to the DR in the Shift_DR state, as shown in Figure 5-9. The data is sent in LSB way or MSB way depending on specific operations.

Figure 5-8 Instruction Register Access Timing
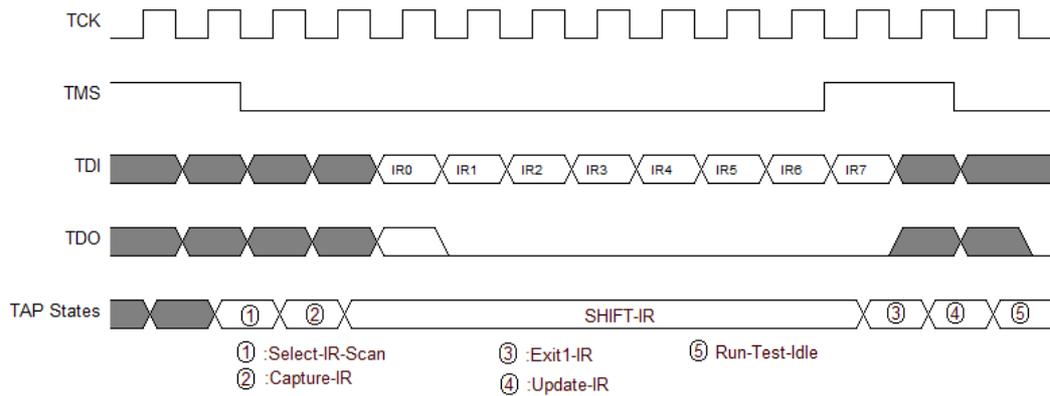
Figure 5-9 Data Register Access Timing
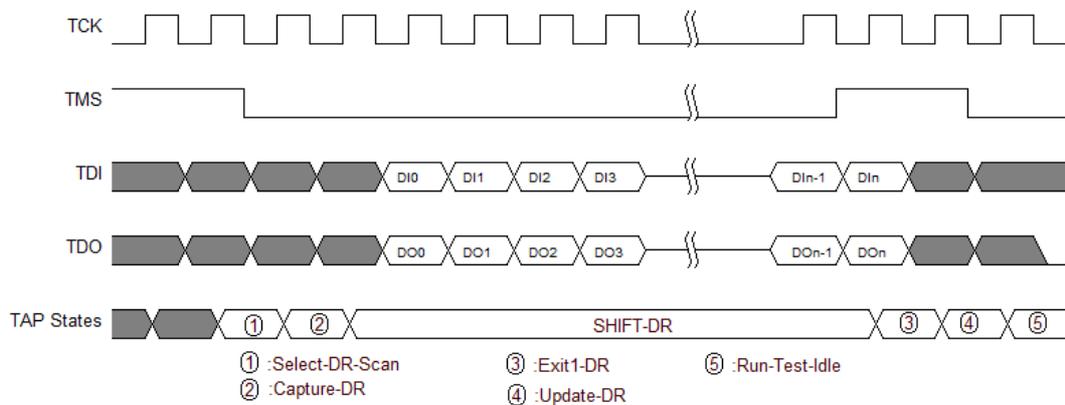
Note!
● The total length of the instruction register is 8 bits in the GW1N(R) and GW2A(R) series of the FPGA;
● The length of the data register can vary depending on the selected register.

Read ID CODE Instance

ID Code, i.e. JEDEC ID Code, is a basic identification of FPGA products.

The length of the Gowin FPGA ID Code is 32 bits. The ID Codes of the FPGA are listed in the following table.

Table 5-5 Gowin FPGA IDCODE

Gowin FPGA Device Family IDCODE			
Device Family	Device Part	Manufacturer ID	IDCODE
	Bits 31-12	Bits 11-0	
		h81B	
GW1N-1	h09002		h0900281B
GW1N-1S	h09003		h0900381B
GW1NZ-1	h01006		h0100681B
GW1NS-2	h03000		h0300081B
GW1NS(R)-2C	h03001		h0300181B
GW1NSE-2C	h03001		h0300181B
GW1N(R)-4	h01001		h0100381B
GW1N(R)-4B	h11003		h1100381B
GW1N(R)-4C	h11003		h0100181B
GW1NS(ER)-4C	h01009		h0100981B
GW1N(R)-9	h11005		h1100581B
GW1N(R)-9C	h11005		h1100481B
GW2A(R)-18/18C	h00000		h0000081B
GW2A-55/55C	h00002	h81B	h0000281B

The instruction for reading FPGA is 0x11. Take the GW1N-4B ID Code as an example to illustrate the working mode of JTAG, please refer to the following steps:

1. TAP reset: TMS is set to high level and at least 5 clock cycles are continuously transmitted;
2. Move the state machine from Test-Logic-Reset to Run-Test-Idle;
3. Move the state machine to Shift-IR. Send Read ID instruction (0x11) beginning with LSB. When MSB (the last bit) is being sent, move state machine to Exit1-IR at the same time, i.e., TMS should be high level before sending MSB. Table 5-6 shows the change of TDI and TMS value during sending 0x11 in 8-clock cycle. The timing is as shown in Figure 5-11.

Table 5-6 Change of TDI and TMS Value in The Process of Sending Instructions

	TCK 1	TCK 2	TCK 3	TCK 4	TCK 5	TCK 6	TCK 7	TCK 8
TDI value (0x11)	1	0	0	0	1	0	0	0
TMS	0	0	0	0	0	0	0	1

	TCK 1	TCK 2	TCK 3	TCK 4	TCK 5	TCK 6	TCK 7	TCK 8
value								

4. Move the state machine, back to Run-Test-Idle after going from Exit1-IR to Update-IR, and then run the state machine at least 3 clock cycles in Run-Test-Idle.

5. Move the state machine to Shift-DR, send 32 clock cycles, and set TMS to high level before the 32nd clock is sent. When the 32 clock cycles are completed, jump from Shift-DR to Exit1-DR. During this period, sending 32 clocks can read 32 bits data, that is, 0x0100381B, as shown in Figure 5-12;

6. Move the state machine back to Run-Test-Idle;

Figure 5-10 Read Machine Flow Chart in ID Code State
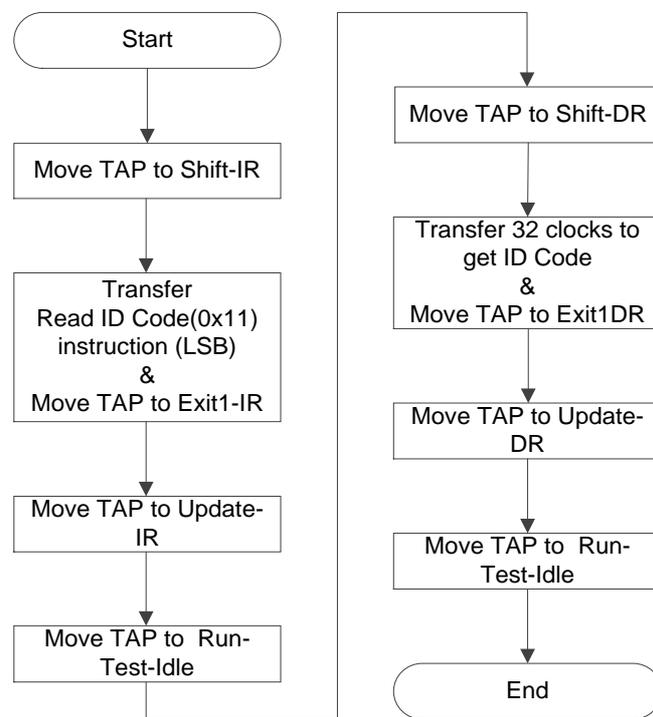
Figure 5-11 The Access Timing of Read ID Code Instruction- 0x11
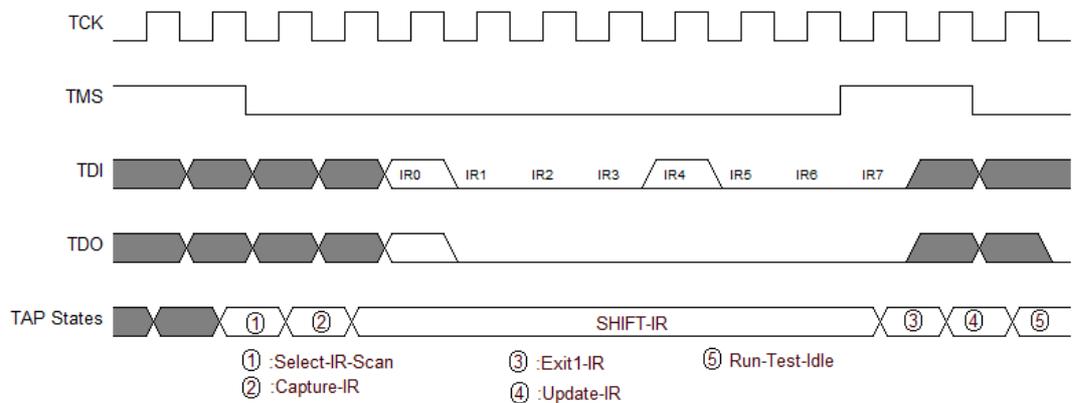
Figure 5-12 Read ID Code Data Register Access Timing
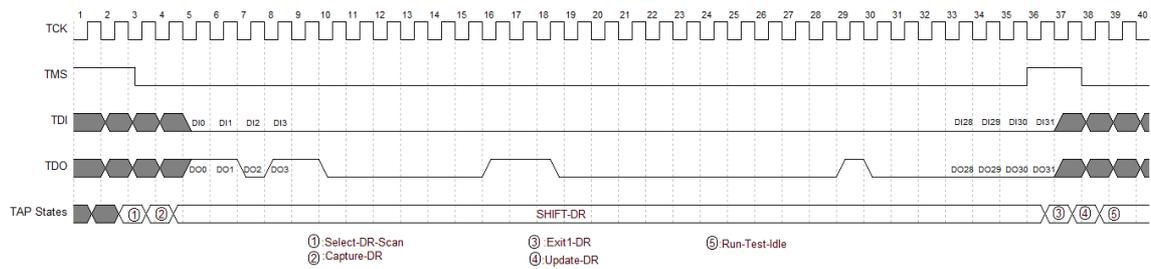
SRAM Configuration Process

    The FPGA SRAM is configured using an external Host to enable the FPGA functions. SRAM is configured via JTAG to avoid the influence of Configuration Mode Pins.

    Generate the FS file using Gowin software. Configure SRAM via JTAG. The process of SRAM configuration using the external Host is as follows, as shown in Figure 5-13.

1. Establish a JTAG link and reset TAP;
2. Read the device ID CODE and check if it matches.
3. Erase the SRAM if it has been configured. Please refer to "SRAM Erasure Process".
4. Send the "0x15" instruction of ConfigEnable;
5. Send the "0x12" instruction of Address Initialize;
6. Send the "0x17" instruction of Transfer Configuration Data.
7. Move the state machine to Shift-DR (Data Register). Send Configuration Data from the MSB bit by bit till all the bitstream file content is sent.
8. Send the "0x3A" instruction of ConfigDisabled;
7. Send the "0x02" instruction of Noop to end the configuration process.
8. Please refer to Process of Reading SRAM (The process of reading SRAM) if reading back Configuration Data is required for verification.

Figure 5-13 SRAM Configuration Flow
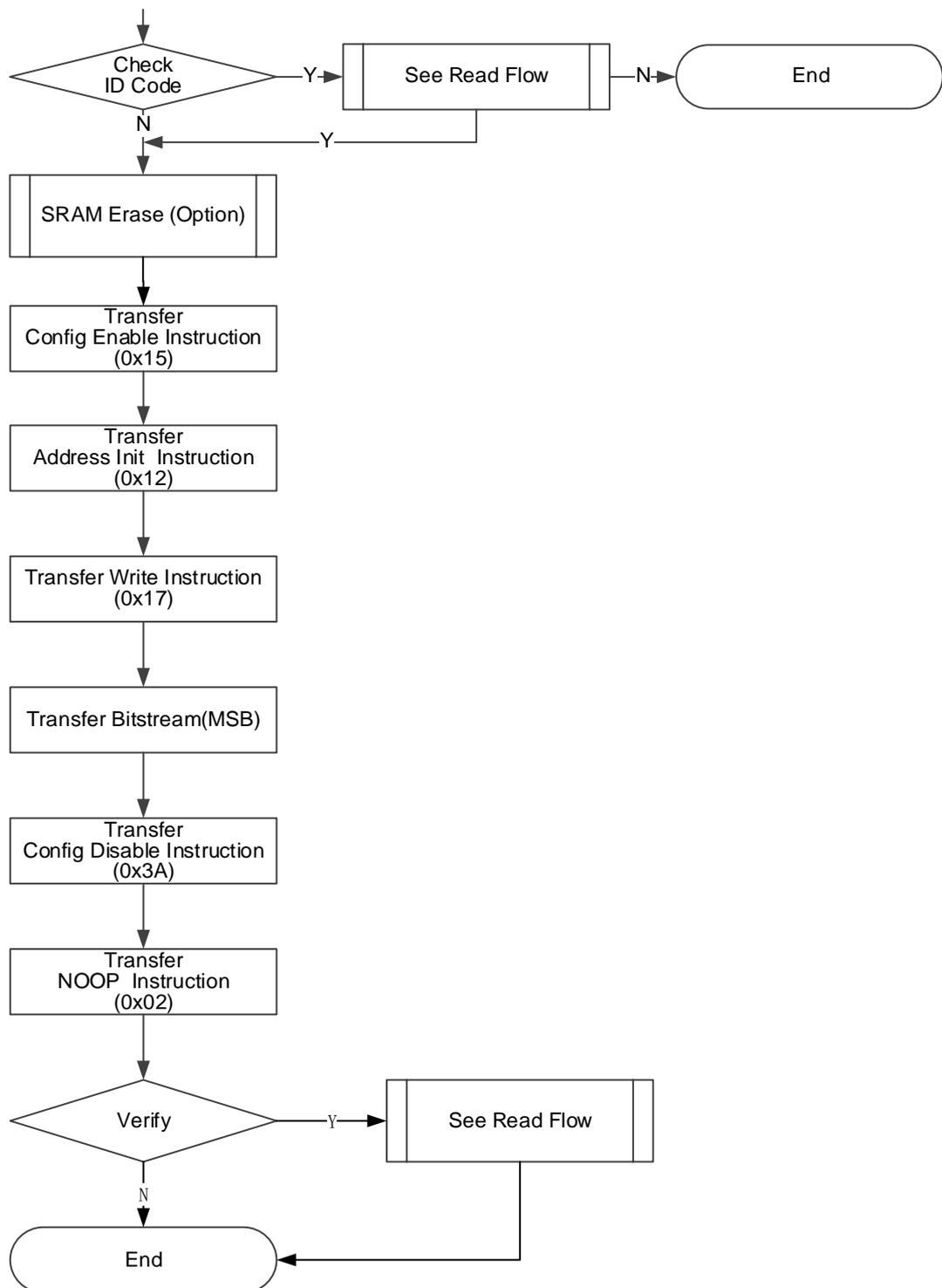
Process of Reading SRAM

Warning: SRAM data is not allowed to be read back by default.
Read the SRAM data from the SRAM area of the FPGA. First ensure that the security bit is not configured when the data are written to the SRAM. The security bit is used to protect the runtime data and ensure the data security. After the safety bit is set, the data received from the SRAM are 1 (high level).

During loading, FPGA performs CRC check on the written data to ensure that the data is written correctly, and whether CRC reports an error can be used as a check mechanism to configure SRAM.

Table 5-7 Count of Address and Length of One Address

Device	Length of One Address (bits/address)	Count of Address
GW1N-1/GW1N-1S/ GW1NZ-1	1216	274
GW1N-2/GW1N(R)-4B/ GW1NS(E/R)-2(C)	2296	494
GW1N(R)-6/GW1N(R)-9	2836	712
GW2A(R)-18	3376	1342
GW2A(R)-55(ES)	5536	2038

The reading process is described in detail below, as shown in Figure 5-14.
1. Send the "0x15" instruction of ConfigEnable;
2. Send the "0x12" instruction of Address Initialize;
3. Send the "0x 03" instruction of SRAM Read;
4. Move the state machine to Shift-DR (data register) and send as many clocks as the value of the address length, see Table 5-7. When the last clock is sent, pull up TMS at the same time. The state machine jumps to Exit1-DR, and TDO reads data with corresponding length. The state machine will return to Run-Test-Idle state finally.
5. Repeat the step 4, the address will be automatically accumulated when the data of an address are read each time;
6. Send the "0x3A" instruction of ConfigDisabled;
7. Send the "0x02" instruction of Noop to end the reading process.

Figure 5-14 Process of reading SRAM

```
                    ┌─────────────┐
                    │    Start    │
                    └─────────────┘
                           │
                           ▼
              ┌──────────────────────────┐
              │         Transfer         │
              │ Config Enable Instruction│
              │          (0x15)          │
              └──────────────────────────┘
                           │
                           ▼
              ┌──────────────────────────┐
              │ Transfer Initialize Address│
              │     Instruction (0x12)   │
              └──────────────────────────┘
                           │
                           ▼
              ┌──────────────────────────┐
              │         Transfer         │
              │     Read Instruction     │
              │          (0x03)          │
              └──────────────────────────┘
                           │
                           ▼
        ┌──────< Next address is valid >──────┐
        │              │ Y                    │ N
        │              ▼                      │
        │   ┌──────────────────────────┐      │
        │   │  Read data of one address│      │
        │   └──────────────────────────┘      │
        │              │                      │
        └──────────────┘                      │
                                              ▼
              ┌──────────────────────────┐
              │        Compute the       │◄─────
              │      checksum(16bit)     │
              └──────────────────────────┘
                           │
                           ▼
              ┌──────────────────────────┐
              │         Transfer         │
              │ Config Disable Instruction│
              │          (0x3A)          │
              └──────────────────────────┘
                           │
                           ▼
                    ┌─────────────┐
                    │     End     │
                    └─────────────┘
```
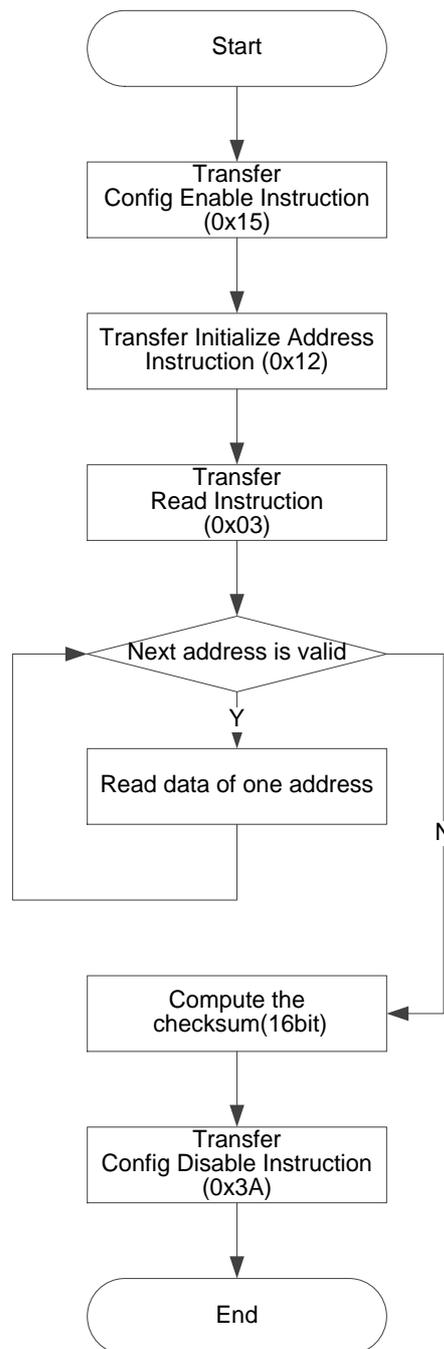
SRAM Erasure Process

When reconfiguring SRAM, the existing SRAM needs to be erased. The flow is as follows:

1. Send the "0x15" instruction of ConfigEnable;
2. Send the "0x05" instruction of SRAM Erase;
3. Send the "0x02 " instruction of Noop;
4. Delay or Run Test 2~10ms;
5. Send the "0x09" instruction of SRAM Erase Done;
6. Send the "0x3A" instruction of ConfigDisabled;

7. Send the "0x02" instruction of Noop to end the Erasure process.

Note!

You need to wait enough time for the device to finish erasing after the instructions of EraseSram(0x05) and Noop(0x02) are sent.

- The reference time for GW1N(*)-1 is 1ms;
- The reference time for GW1N(*)-4 is 2ms;
- The reference time for GW1N(*)-9 is 4ms;
- The reference time for GW2A(*)-18 is 6ms;
- The reference time for GW2A(*)-55 is 10ms.

Erase Internal Flash

For the embedded Flash memory of GW1N series, the embedded Flash needs to be erased before each programming task. For data security, the embedded Flash must be erased entirely.

The requirements for JTAG programming frequency are different according to the different processes of the GW1N series of the embedded Flash. Please refer to Table 5-8.

Table 5-8 TCK Frequency Requirements for JTAG

Device	TCK Frequency Range	Process Code
GW1N-1 GW1N-1S	1.4MHz ~ 5MHz	H
GW1N(RF)-4B GW1N(SER)-4C GW1N(R)-9(C) GW1NZ-1	1MHz ~ 5MHz	T
GW1NS(E)-2(C)	1MHz ~ 5MHz	S

FPGA erasure process of T Technology

The following describes the erase flow of T Technology for GW1NZ-1 in detail, as shown in Figure 5-15.

1. Establish a JTAG link and reset the TAP;
2. Read the device ID CODE and check if it matches.
3. Erase SRAM first if it has been configured.
4. Send the "0x15" instruction of ConfigEnable;
5. Send the "0x75" instruction of EFlash Erase;
6. The clock（Run-Test） is continuously generated in Run-Test-Idle for 500µs;
7. Move the state machine in turn: Run-Test-Idle -> Select-DR-Scan-> Update-DR -> Capture-DR -> Shift-DR -> Transfer 32 bits-> Exit1-DR -> Update-DR -> Run-Test-Idle;
8. The clock（Run-Test）is continuously generated in Run-Test-Idle for 120ms. Please refer to Table 5-8 for the frequency requirements;
9. Send the "0x3A" instruction of ConfigDisabled;
10. Send the "0x02" instruction of Noop to end the erasure process.
11. Send the "0x03" instruction of Reprogram to reconfigure the device and check if it erases successfully.

Figure 5-15 The Embedded Flash Erasing process of T Technology
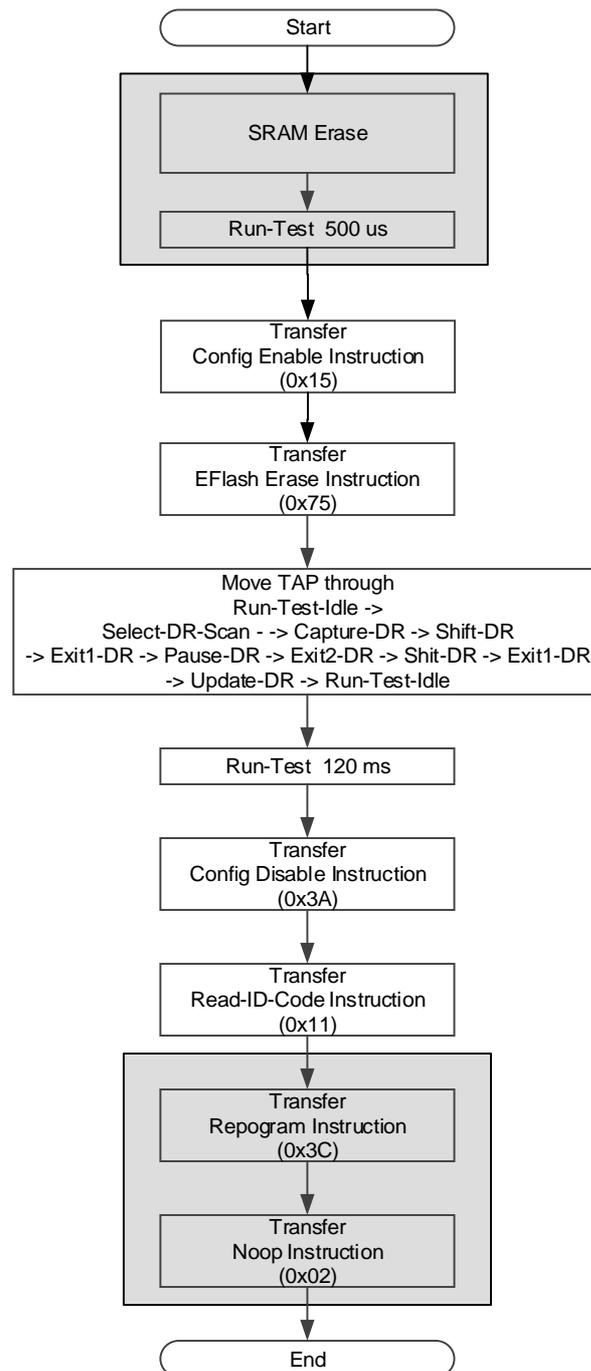
Note!

Ignore the shading area operation during Background Programming.

FPGA erasure process of H Technology

FPGA erasure process of H Technology:

1. Send the "0x15" instruction of ConfigEnable;
2. Send the "0x75" instruction of EFlash Erase;
3. Move the state machine from Run-Test-Idle to Shift-DR; 32 clocks are generated (TDI signal keeps low level). Move the state machine to Exit1-DR at the 32th clock, and then return to Run-Test-Idle going from

Update-DR;
4. Repeat the steps above, 65 times in all;
5. The clock（Run-Test）is continuously generated in Run-Test-Idle for 120ms. Please refer to Table 5-8 for the frequency requirements;
6. Send the "0x3A" instruction of ConfigDisabled;
7. Send the "0x03" instruction of Reprogram to check if the erasing is successful;
8. Send the "0x02" instruction of Noop to end the erasure process.

Figure 5-16 The Embedded Flash Erasing process of S Technology

```
                    ┌──────────────────┐
                    │      Start        │
                    └──────────────────┘
                             │
                             ▼
                    ┌──────────────────┐
                    │     Transfer      │
                    │ Config Enable Instruction │
                    │      (0x15)       │
                    └──────────────────┘
                             │
         ┌───────────────────▼────────────────────┐
         │          ┌──────────────────┐          │
         │          │     Transfer      │          │
         │          │ SRAM Erase Instruction │     │
         │          │      (0x05)       │          │
         │          └──────────────────┘          │
         │                   │                     │
         │                   ▼                     │
         │          ┌──────────────────┐          │
         │          │   Run-Test 1ms    │          │
         │          └──────────────────┘          │
         │                   │                     │
         │                   ▼                     │
         │          ┌──────────────────┐          │
         │          │     Transfer      │          │
         │          │ SRAM Erase Done Instruction │ │
         │          │      (0x09)       │          │
         │          └──────────────────┘          │
         │                   │                     │
         │                   ▼                     │
         │          ┌──────────────────┐          │
         │          │  Run-Test 500 us  │          │
         │          └──────────────────┘          │
         └───────────────────┬────────────────────┘
                             │
                             ▼
                    ┌──────────────────┐
                    │     Transfer      │
                    │ EFlash Erase Instruction │
                    │      (0x75)       │
                    └──────────────────┘
                             │
                             ▼
          ┌────────────────────────────────────────┐
          │            Repeat 65 times:            │
          │            Run-Test-Idle ->            │
          │ Select-DR-Scan -> Update-DR -> Capture-DR -> Shift-DR │
          │         -> Transfer 32 bits -> Exit1-DR │
          │         -> Update-DR -> Run-Test-Idle   │
          └────────────────────────────────────────┘
                             │
                             ▼
                    ┌──────────────────┐
                    │   Run-Test 96 ms  │
                    └──────────────────┘
                             │
                             ▼
                    ┌──────────────────┐
                    │     Transfer      │
                    │ Config Disable Instruction │
                    │      (0x3A)       │
                    └──────────────────┘
                             │
                             ▼
                    ┌──────────────────┐
                    │     Transfer      │
                    │  Noop Instruction │
                    │      (0x02)       │
                    └──────────────────┘
                             │
                             ▼
                    ┌──────────────────┐
                    │       End         │
                    └──────────────────┘
```

GW1NS(E)-2(C) Erasure Process of S Technology

GW1NS(E)-2(C) offers two built-in Flash. Note the different Flash when programming. Refer to the process below:
1. Check if the device ID is matched;
2. Send the "0x15" instruction of ConfigEnable;
3. If the second Flash needs to be erased, send the "0x78" instruction of Flash 2nd Enable.

Note!

The condition to erase the second Flash is that FPGA should be in the Wakeup state. (Done Final of Status Code should be 1.)

4. Send the "0x75" instruction of EFlash Erase;
5. Move the state machine to Shift-DR, and generate a clock of 110ms. Please refer to Table 5-8 for the frequency requirements. Return to Run-Test-Idle;
6. Send the "0x3A" instruction of ConfigDisabled;
7. Send the "0x02" instruction of Noop to end the process.

Process of Programming Internal Flash

The internal Flash uses 256Bytes as an X-page. Each X-page is divided into 64 Y-pages, and each Y-page contains 4Bytes.

The first Y-page of the first X-page is used to identify whether the Flash has the capability of Autoboot or Readback, as shown in Table 5-9. When Readable-pattern is written to the first Y-page, the Flash data can be read; when the Autoboot-pattern is written to the first Y-page, the device automatically loads the Flash data into the SRAM in the autoboot mode; The Flash can only be read after the Readable-pattern is written. It cannot be read in any other cases. Devices with the feature of background programming just need to use Autoboot-pattern.

Autoboot-pattern data must be inserted in the header of bitstream file in the case of no requirements of reading back data. If an X-page is less than 256Bytes, you can use 0xFF or 0x00 to complement it.

The requirements for JTAG programming frequency are different according to the different processes of the embedded Flash in GW1N series. Please refer to SRAM Erasure Process> Table 5-8 TCK Frequency Requirements for JTAG.

Table 5-9 Readback-pattern / Autoboot-pattern

Device	Readable-pattern(4 Bytes)	Autoboot-pattern(4 Bytes)
GW1N-1/GW1N-1S	0x07,0x07,0x30,0x40	0x47,0x57,0x31,0x4E
GW1N(R)-2/4 GW1N(R)-2B/4B/9 GW1NZ-1 GW1NS(E)-2(C)	0xF7,0xF7,0x3F,0x4F	

The process of programming internal Flash is shown in :

1. Check whether the ID Code matches;
2. Erase the embedded Flash;
3. Verify if the erasure is successful by reading the Status register to check if the device has been restored to the initial state of the die; the background programming devices and the GW1NS series of devices cannot be checked by reading the Status register;
4. Send the "0x15" instruction of ConfigEnable;
5. Write one X-page at a time until the programming is completed;
6. Send the "0x3A" instruction of ConfigDisabled;
7. Send a Reprogram instruction (0x3C) to load the Flash data into the SRAM;

8. Read Status Code/User Code to check if the loading is successful.

Figure 5-17 Process of Programming Internal Flash View
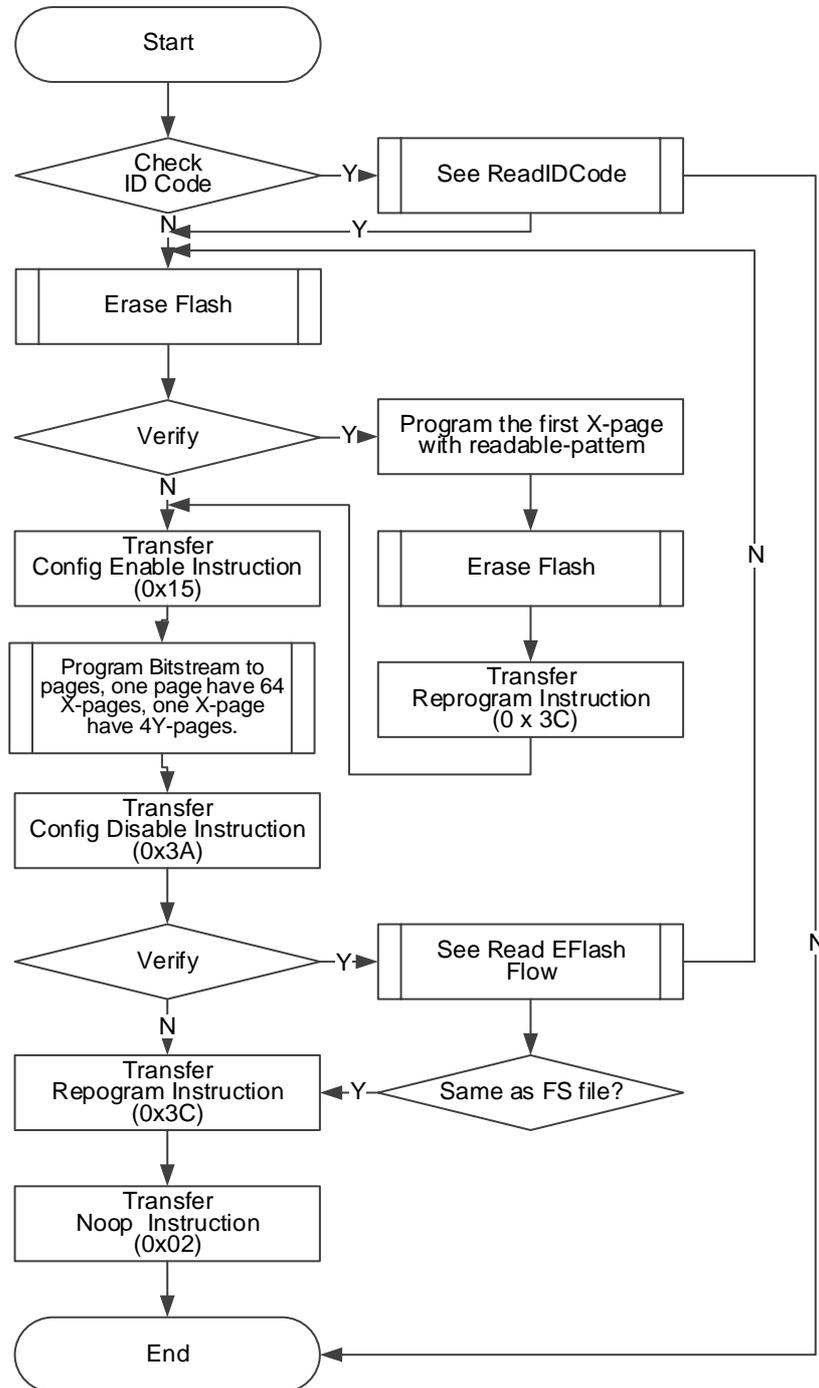
Process of Programming an X-page

The process of programming an X-page is as shown in Figure 5-18.

1. Send the "0x15" instruction of ConfigEnable;
2. Send the "0x71" instruction of EF-Program;
3. Enter into Shift-DR and send address data[1];
4. Write an X-page data.

One Y-page has 256 bytes in all. Program 4 Bytes and program 64 times for one Y-page. The Y-page data is written in LSB way. Refer to Figure 2-15.

5. After one X-page is written in, GW1N-1(S) needs to perform Run-Test for 2400μs; GW1N(Z)-2/4/6/9 needs to perform Run-Test for 6μs. No extra clock is required for the other series of devices.

6. This X-page programming ends.

Note!

Address data format is 32 bits altogether, and the lower 6 bits are reserved. For example, when the address is b'**00010011**（0x13）, the written-in address is **00000000000000000000010011**000000. The address data is written in LSB way. Jump out of Shift-DR at the last bit.

Figure 5-18 X-page Programming

```
                    ┌──────────────┐
                    │    Start     │
                    └──────┬───────┘
                           │
              ┌────────────▼─────────────┐
              │  Transfer Config-Enable  │
              │   Instuction (0x15)      │
              └────────────┬─────────────┘
                           │
              ┌────────────▼─────────────┐
              │   Transfer EF-Program    │
              │   Instuction (0x71)      │
              └────────────┬─────────────┘
                           │
                     ┌─────▼──────┐        ┌──────────────────────┐
                     │  Address   │───Y───▶│ Delay 16000ns in     │
                     │ index > 0  │        │ Run-Test-Idle        │
                     └─────┬──────┘        └──────────────────────┘
                           │◀────────N───────────────┘
              ┌────────────▼─────────────┐
              │ Transfer address data (LSB) │
              └────────────┬─────────────┘
                           │
              ┌────────────▼─────────────┐
              │     Delay 16000ns        │
              └────────────┬─────────────┘
                           │
              ┌────────────▼─────────────┐
              │     Program 1 X-Page     │
              └────────────┬─────────────┘
                           │
              ┌────────────▼─────────────┐
              │          Delay           │
              │  6μS (GW1N(Z)-2/4/6/9)   │
              │           Or             │
              │  2400μS (GW1N-1(S))      │
              │    in Run-Test-Idle      │
              └────────────┬─────────────┘
                           │
                    ┌──────▼───────┐
                    │     End      │
                    └──────────────┘
```
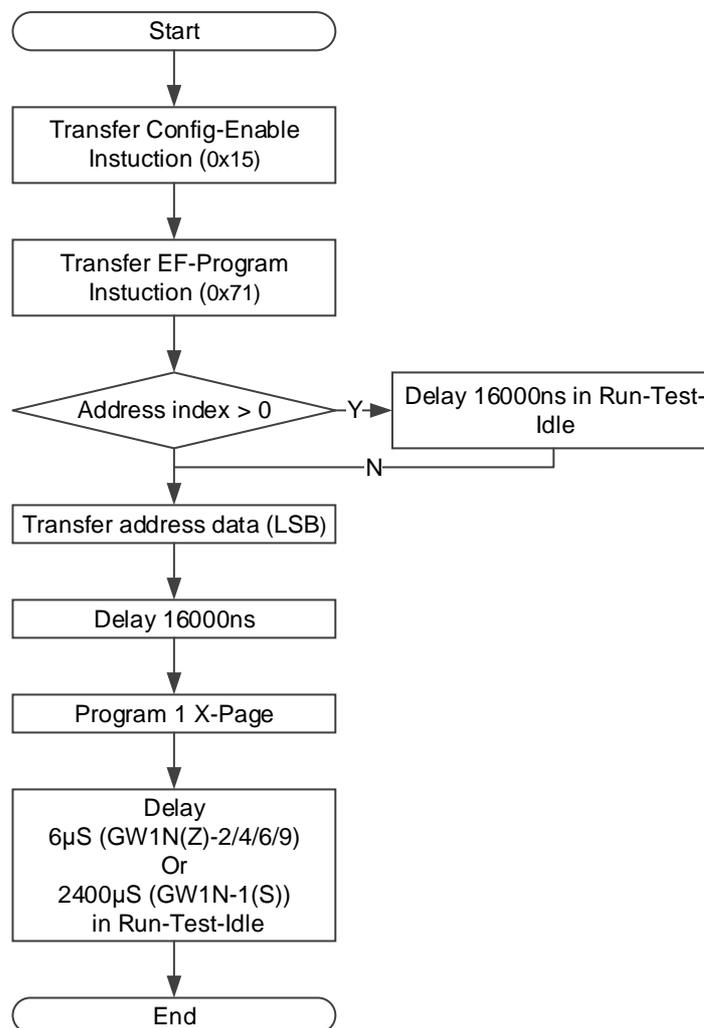
Process of Programming an Y-page

Y-page programming is the smallest unit in programming process. 4 Bytes are written each time in the LSB way, as shown in Figure 5-19.
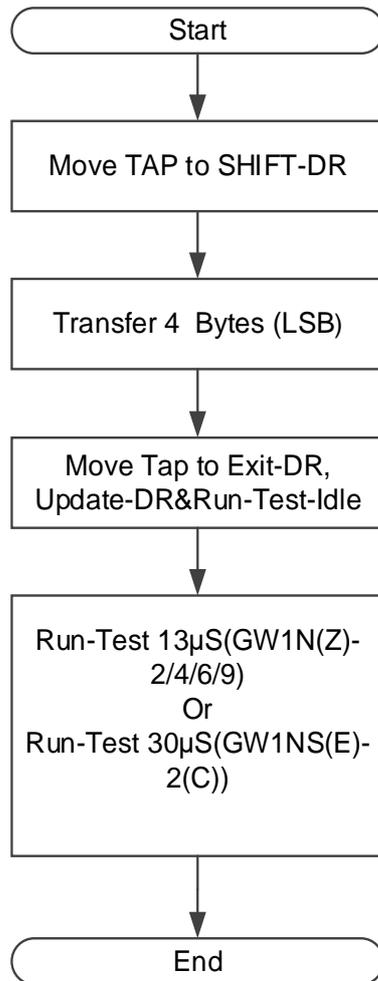
Different series of devices all need to perform Run-Test to wait for writing all Bytes, and the JTAG clock needs to meet minimum frequency requirements. Refer to Table 5-8.

After one Y-page is written in each time, GW1N(Z)-2/4/6/9 needs to perform Run-Test for 13-15 µs; GW1N(S)-2(C) needs to perform Run-Test for 30-35 µs. No extra clock is required for the other series of devices.

Note!

If you want to read data from Configuration Data, high 4 Bytes will be taken. If you want to write data into Shift-DR, LSB will begin to write.

Figure 5-19 Y-page Programming

```
         ┌────────────────┐
         │     Start       │
         └────────┬────────┘
                  │
                  ▼
    ┌─────────────────────────┐
    │  Move TAP to SHIFT-DR    │
    └────────────┬────────────┘
                 │
                 ▼
    ┌─────────────────────────┐
    │  Transfer 4  Bytes (LSB) │
    └────────────┬────────────┘
                 │
                 ▼
    ┌─────────────────────────┐
    │  Move Tap to Exit-DR,    │
    │ Update-DR&Run-Test-Idle  │
    └────────────┬────────────┘
                 │
                 ▼
    ┌─────────────────────────┐
    │  Run-Test 13µS(GW1N(Z)-  │
    │         2/4/6/9)          │
    │           Or              │
    │  Run-Test 30µS(GW1NS(E)-  │
    │         2(C))             │
    └────────────┬────────────┘
                 │
                 ▼
         ┌────────────────┐
         │      End        │
         └────────────────┘
```

Process of Reading internal Flash

This chapter introduces the process of reading internal Flash briefly, no rate requirements for the TCK of JTAG, as shown in Figure 5-20.

Reading the internal Flash can be regarded as the reverse process of programming Flash. But firstly, you should make sure that the written-in Readable-pattern has taken effect. For GW1N, the Reprogram(0x3C) and Noop(0x02) can be sent in turn after Readable-pattern is written-in to make the internal flash be Readable.

Process Description:
1. Check ID Code. (optional);
2. Send the "0x15" instruction of ConfigEnable;
3. Send EF-Read instruction 0x73;
4. Send read Flash start address 0x0. The method is same as write

X-address in 0;
5. 64 Y-page is an X-page;
6. After reading one X-page, need not to send address again. The address will recurse automatically;
7. After reading, send the "0x3A" instruction of ConfigEnable to end the process.
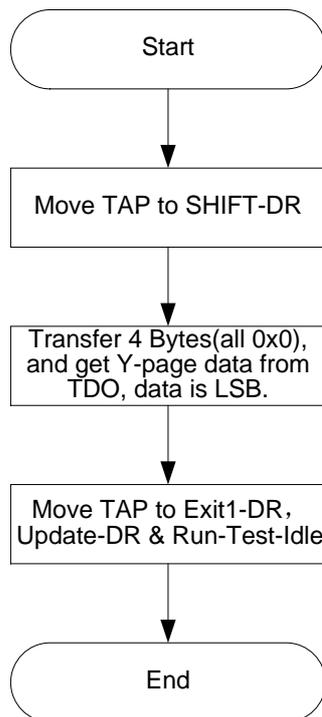
Figure 5-20 Process of Reading Internal Flash
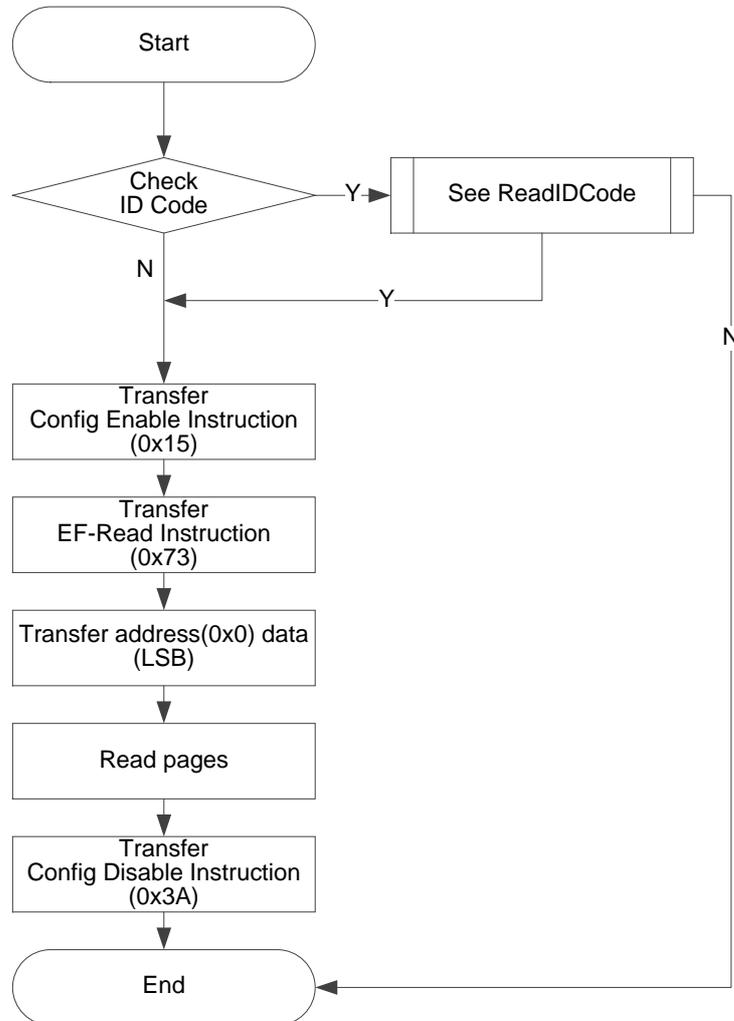
Process of Reading a Page (Y-page) Flash

Reading a Y-page is similar to writing a Y-page, but there is no waiting time for writing in Flash. As shown in Figure 5-21.

The lowest bit in the data is outputted first.

Figure 5-21 Process of Reading a Y-page

```mermaid
flowchart TD
    A([Start]) --> B[Move TAP to SHIFT-DR]
    B --> C[Transfer 4 Bytes all 0x0 and get Y-page data from TDO data is LSB]
    C --> D[Move TAP to Exit1-DR Update-DR Run-Test-Idle]
    D --> E([End])
```

Start

Move TAP to SHIFT-DR

Transfer 4 Bytes(all 0x0), and get Y-page data from TDO, data is LSB.

Move TAP to Exit1-DR，Update-DR & Run-Test-Idle

End

Background Programming

The device sometimes needs to upgrade the data file and program the Flash without affecting current functions. And it can maintain the I/O state when adding a new data stream file. The following is the flow of GW1N4 that upgrades the internal Flash data using the Background Programming.

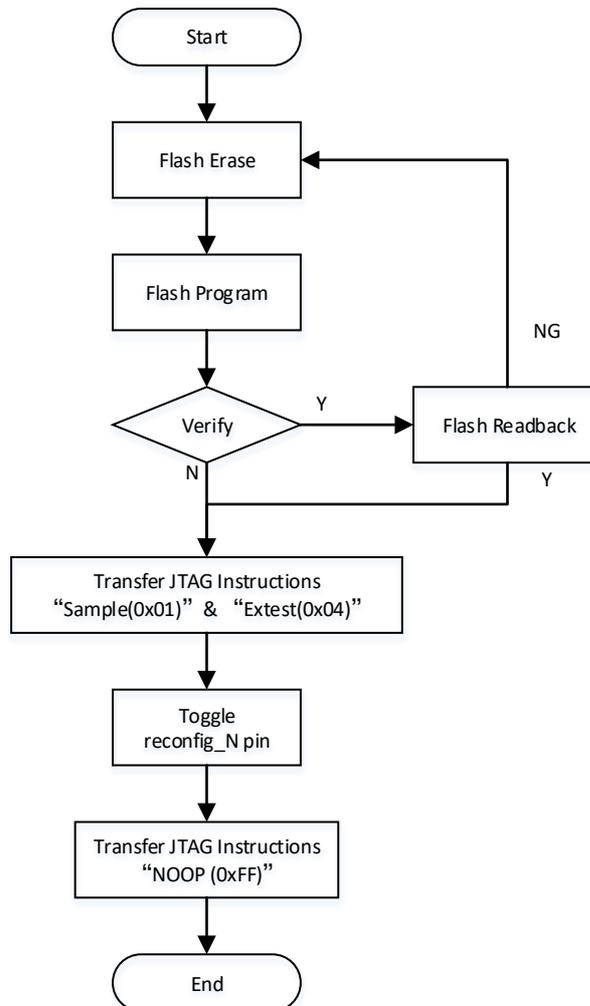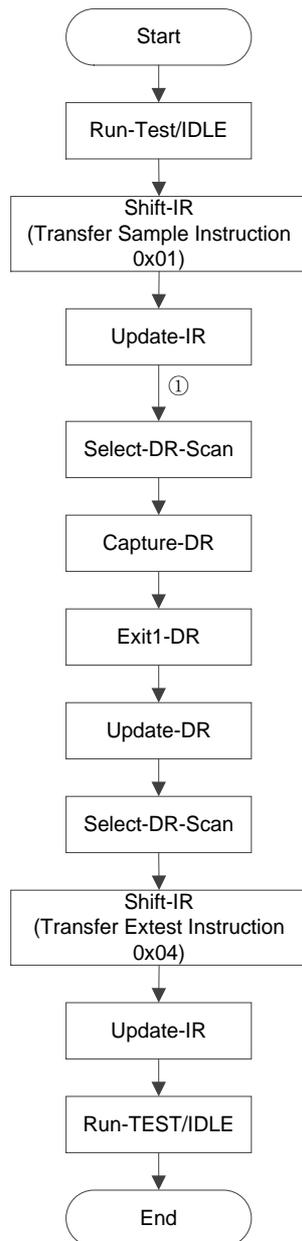**Figure 5-22 GW1N-4 Background Programming Flow**

Figure 5-23 Transfer JTAG Instruction Sample & Extest Flow Chart

```
                    ( Start )
                        │
                        ▼
              ┌──────────────────┐
              │   Run-Test/IDLE   │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────────────┐
              │        Shift-IR           │
              │ (Transfer Sample Instruction│
              │          0x01)            │
              └──────────────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │     Update-IR     │
              └──────────────────┘
                        │ ①
                        ▼
              ┌──────────────────┐
              │  Select-DR-Scan   │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │    Capture-DR     │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │     Exit1-DR      │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │    Update-DR      │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │  Select-DR-Scan   │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────────────┐
              │        Shift-IR           │
              │ (Transfer Extest Instruction│
              │          0x04)            │
              └──────────────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │     Update-IR     │
              └──────────────────┘
                        │
                        ▼
              ┌──────────────────┐
              │   Run-TEST/IDLE   │
              └──────────────────┘
                        │
                        ▼
                    ( End )
```

Note!

① Jump directly from Update-IR to Select-DR-Scan.

ExFlash Programming

Gowin FPGA can load bitstream files from external Flash and program external Flash through JTAG directly.

Figure 5-24 Connection Diagram of JTAG Programming External Flash
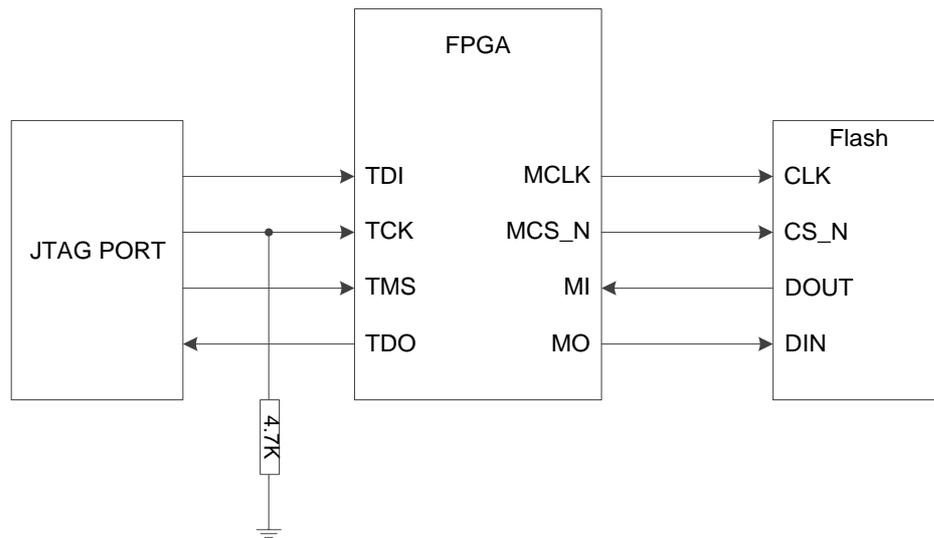
Note!

The figure above shows the minimum system diagram of programming external Flash via JTAG.

Program External Flash via JTAG-SPI

In this mode, the external Flash can be programed via JTAG.

The principle of this mode is to convert JTAG protocol to SPI protocol and then program external Flash. Users program SPI Flash by simulating Master SPI timing through JTAG.
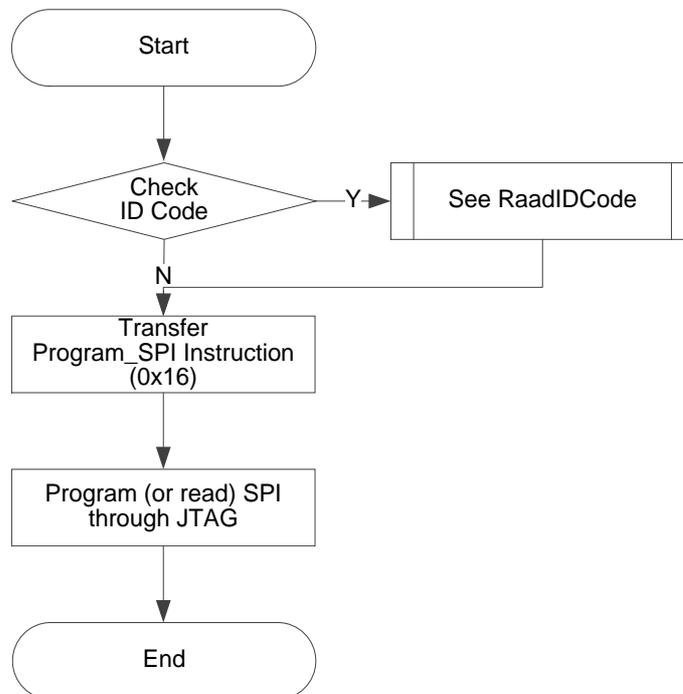
Figure 5-25 Process View of Programming SPI Flash SPI

Figure 5-26 Timing Diagram of Sending 0x06 via GW2A series JTAG Simulating SPI

1 Run-Test/Idle
2 Select-DR-Scan
3 Capture-DR
4 Shift-DR
5 Exit1-DR
6 Pause-DR
7 Exit2-DR
8 Update-DR
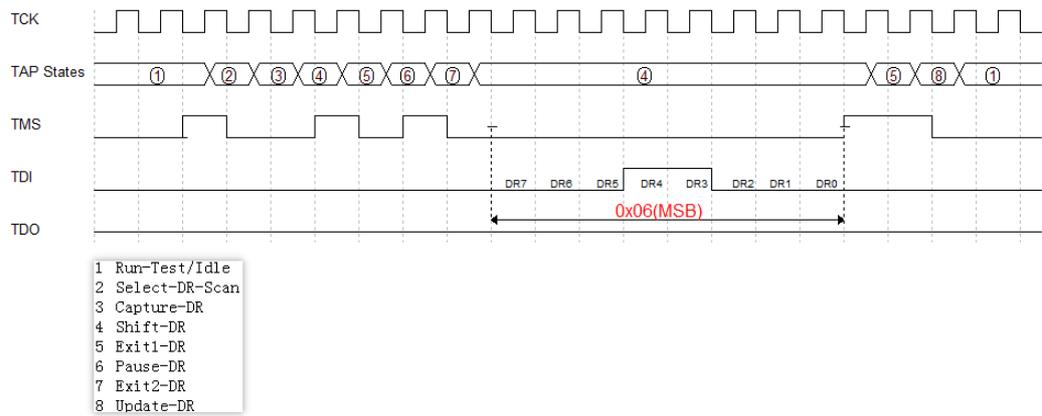
Figure 5-27 Timing Diagram of Sending 0x06 via GW1N series JTAG Simulating SPI

1 Run-Test/Idle
2 Select-DR-Scan
3 Capture-DR
4 Shift-DR
5 Exit1-DR
6 Pause-DR
7 Exit2-DR
8 Update-DR
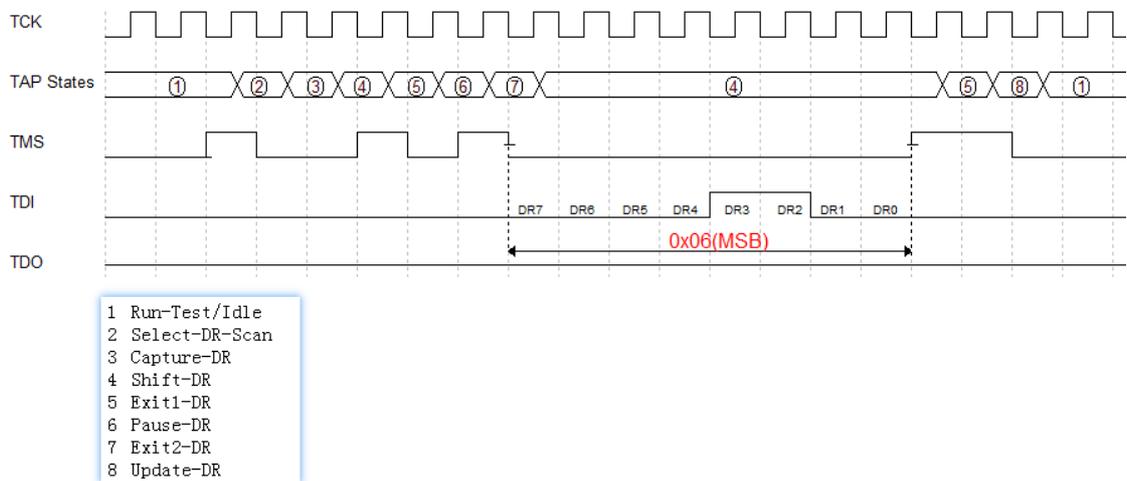
Program SPI Flash in JTAG Boundary Scan Mode

The principle of this mode is changing the state of the pins connected to SPI by using Boundary Scan method to implement SSPI timing, and then to program the internal Flash.

The length of the Boundary Scan Chain used in this mode is 8 bits. Every two bits combination corresponds to the pin state, as shown in Table 5-10. One SCLK drive is completed every two times of sending Boundary Scan Chain.

Table 5-10 Pin State

Pins Name of SPI Flash	SCLK		CS		DI		DO	
Bscan Chain[7:0]	7	6	5	4	3	2	1	0
(ctrl & data)	0		0		0		1	

Note!
- ctrl:0 means output, 1 means input;
- data:0 means low, 1 means high.

Figure 5-28 Process of Use Boundary Scan Mode To Program SPI Flash
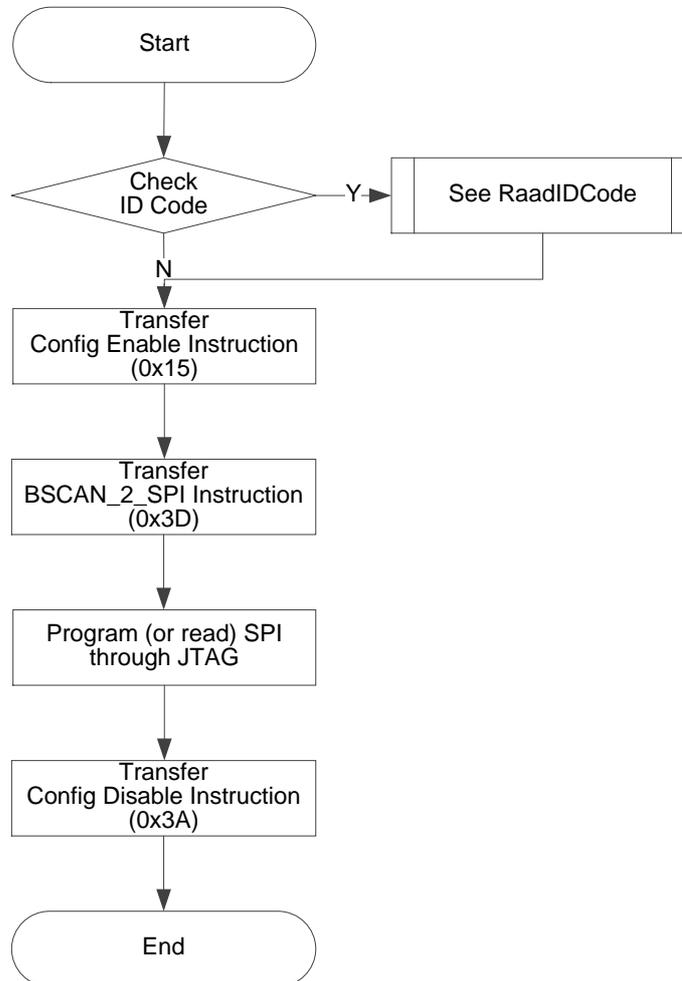
Read Status Register 0x41

Status Register is of great help in device debugging and observing device Status. Reading Status Register can preliminarily judge the Status of devices, such as whether wakeup is successful or not, whether there is a loading error, etc.

The Status Register is 32 bits, the read instruction is 0x41 and the timing is the same as that of Read ID Code.

The meaning of the Status Register is shown in Table 5-11.

Table 5-11 Status Register Definition

Device Status Register[31:0]	GW1N(R)-1/2/4	GW1NS-2 GW1NS(R)-2C	GW1N(R)-6/9 GW1NZ-1	GW2A-18/55
0	CRC Error			
1	Bad Command Error			
2	ID Verify Failed Error			
3	Timeout Error			
4	0			
5	-			
6	-			
7	-			
8	-			
9	0			
10	-			
11	-			
12	Gowin VLD(1)			0
13	Done Final			
14	Security Final			
15	Ready(1)	Ready(0)	Ready(1)	Encrypted format
16	POR(1)			Encrypted key is right
17	0	-	-	0
18	0	-	0	0
19-31	0			

* Gowin VLD is associated with the embedded Flash.

Read Code 0x13

The user code is 32 bits, the read instruction is 0x13 and the timing is the same as that of Read ID Code.

The user code adopts the checksum value in the FS file by default. It can be redefined using Gowin Designer.
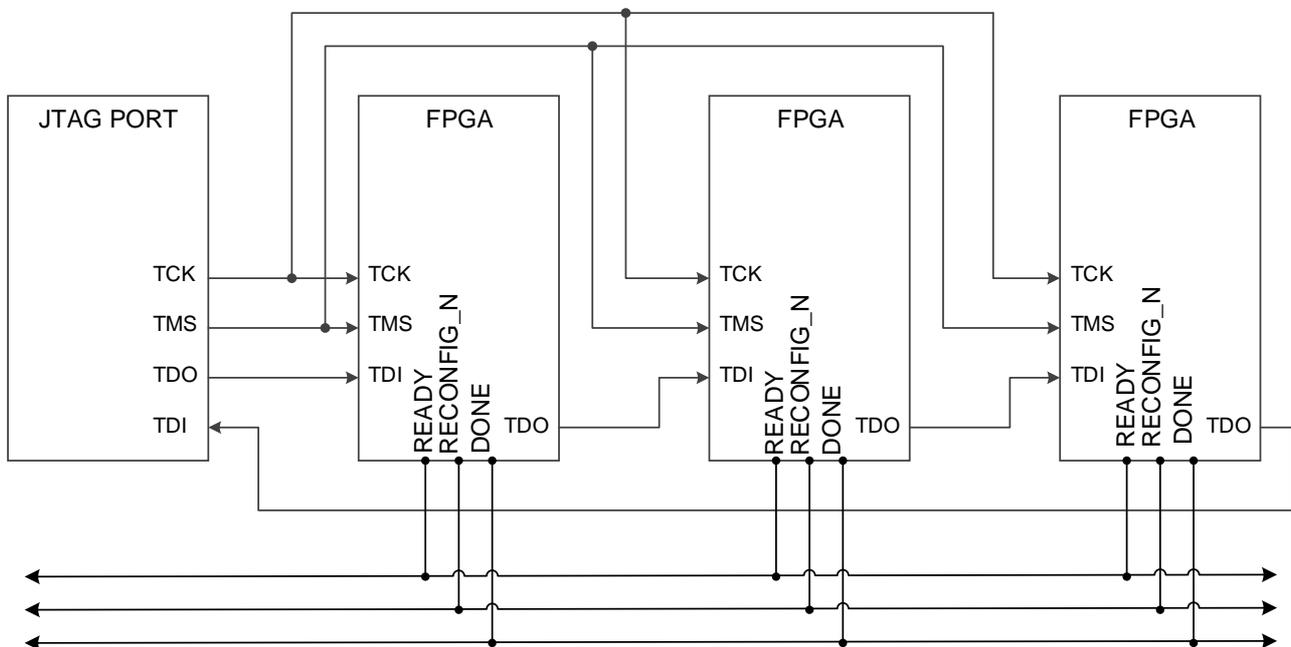
Reload 0x3C

This instruction is used to read the bitstream files from Flash and write to SRAM.

Send the instructions of Reprogram (0x3C) and Noop (0x02) to reload the device via JTAG. You can also reload the device by triggering the Reconfig_N pin.

Connection Diagram of Daisy-Chain

Figure 5-29 Connection Diagram of Daisy-Chain

Routine File

For the routine file, please contact GOWINSEMI technical support or the local office.

5.3 AUTO BOOT Configuration (Supported by LittleBee® Family Only)

The AUTO BOOT mode is a configuration mode for momentary connection feature of non-volatile LittleBee® family of FPGA Products. The Arora Family of FPGA products do not support AUTO BOOT mode. In AUTO BOOT mode, FPGA reads bitstream data from the built-in Flash automatically after it is powered on, with no connection to an external configuration port.

In the AUTO BOOT mode, the bitstream data needs to be written to

the built-in Flash via the JTAG port first (refer to Figure 5-4 Connection Diagram for JTAG Configuration Mode), and then set the MODE value to "000", the chip will automatically read the bitstream data to complete configuration when powered up again or RECONFIG_N triggered at a low-level pulse. When the MODE value is set to "000", the FPGA will automatically configure the SRAM to complete AUTO BOOT after the built-in Flash is programmed using Gowin programmer. The momentary connection feature of the built-in Flash saves download time and improves productivity.

GW1N(R) - 9 and GW1NS series support two retries of AUTO BOOT configuration, i.e. the devices can be automatically reconfigured twice if the first configuration fails after power up. The other devices of LittleBee® only support one-time AUTO BOOT configuration. The factors that can lead to a failed configuration include false ID validation, false CRC check, and false instruction.

Note!

The embedded Flash can only store one bitsteam file. The retry address configuration could not be changed

5.4 SSPI

In SSPI (Slave SPI) mode, FPGA is as a slave device and is configured via SPI by an external Host.

5.4.1 SSPI Mode Pins

The SSPI configuration pins are shown in Table 5-12.

Table 5-12 SSPI Mode Pins

Pin Name	I/O	Description
RECONFIG_N	I, Internal weak pull-up	Low level pulse: Start GowinCONFIG
READY	I/O	High level: FPGA can be programmed and configured Low level: Programming configuration for FPGA is prohibited
DONE	I/O	High-level pulse: Successfully programmed and configured; Low-level pulse: Programming and configuration uncompleted or failed.
MODE[2:0]	I, Internal weak pull-up	Configuration mode selection, READY rising edge sampling
SCLK	I	Input clock
CLKHOLD_N	I, Internal weak pull-up	High level: SPI operation corresponding to SCLK is valid Low level: SPI operation corresponding to SCLK is invalid
SO	O	FPGA outputs data to Host
SI	I	Input data to FPGA from Host
SSPI_CS_N	I, Internal weak pull-up	SSPI Chip selection signal, active low.

5.4.2 SSPI Configuration Timing

See Figure 5-30 for the SSPI timing.

Figure 5-30 SSPI Configuration Timing
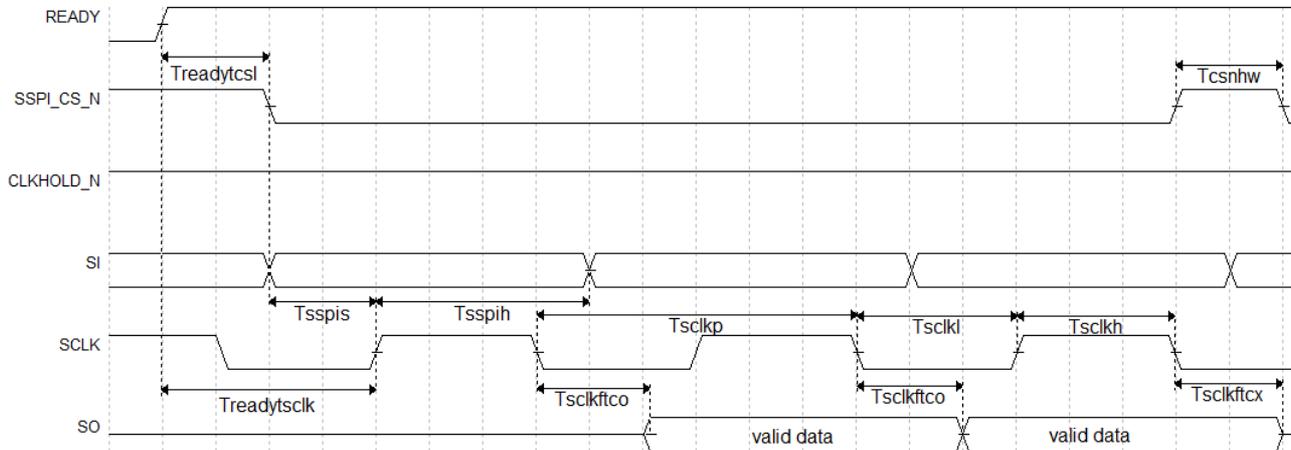
See Table 5-13 for the SSPI configuration timing parameters.

Table 5-13 SSPI Configuration Timing Parameters

Name	Description	Min.	Max.
T_{sclkp}	SCLK clock period	15ns	-
T_{sclkh}	SCLK clock high time	7.5ns	-
T_{sclkl}	SCLK clock low time	7.5ns	-
T_{sspis}	SSPI PORT setup time	2ns	-
T_{sspih}	SSPI PORT hold time	0ns	-
$T_{sclkftco}$	Time from SCLK falling edge to output	-	10ns
$T_{sclkftcx}$	Time from SCLK falling edge to high impedance	-	10ns
T_{csnhw}	CSN high time	25ns	-
$T_{readytcsl}$	Time from READY rising edge to CSN low	TBD	
$T_{readytsclk}$	Time from READY rising edge to first SCLK edge	TBD	-

Other than the power requirements, the following conditions need to be met to use the SSPI configuration mode:

- SSPI port enable
  RECONFIG_N is not set as a GPIO during the first configuration after power up or the previous programming.
- Initiate new configuration
  Power up again or trigger RECONFIG_N at one low pulse.

5.4.3 Configuration Instruction

In Slave SPI mode, you can program FPGA SRAM or read ID information on ID CODE\USER CODE\STATUS CODE through SSPI.. External memory can also be programmed (Such as SPI Flash).

The SSPI instruction of FPGA is generally composed of 1-4 bytes,

including at least 1 instruction class byte and multiple redundant information bytes. If there is no specified information byte, the redundant information byte can be any number (0x00 is used in the following table).

**Table 5-14 Configuration Instruction**

Name	Complete Instruction (Instruction Byte + Redundant Information Byte)
Read ID Code	0x11000000
Read User Code	0x13000000
Read Status Code	0x41000000
Reconfig/Reprogram	0x3C00
Write Enable	0x1500
Write Disable	0x3A00
Write Data	0x3B
Program SPI Flash	0x1600
Init Address	0x1200
Erase SRAM	0x0500

Read ID Code

The length of FPGA ID Code is 32bits. The instruction to read ID is four bytes, that is 0x11000000. Before sending instructions, keep CS at a high level and generate multiple clocks (more than two) to let FPGA get CS state.

After CS is pulled down, the instruction of 0x11000000 is written in in MSB way and after this, 32 clocks are generated continuously. At this time, the ID CODE data will be successively shifted out of DO in the form of MSB.

Figure 5-31 Read ID Code Timing
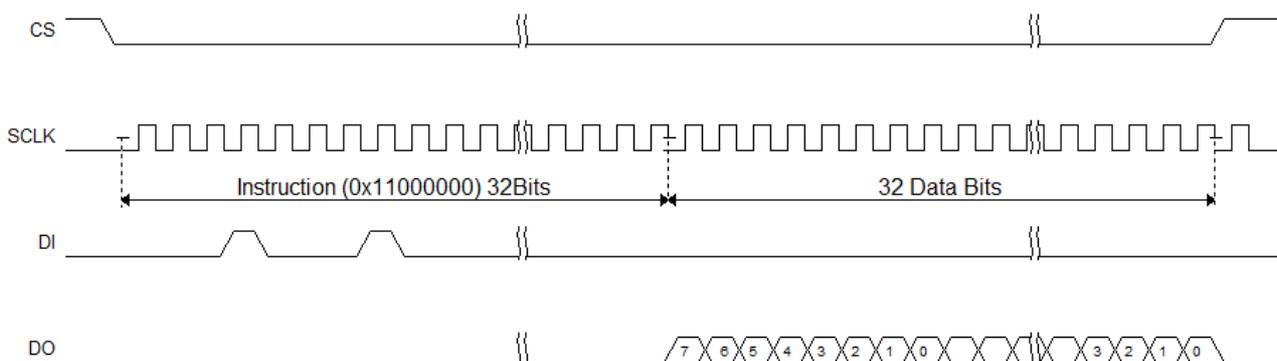
The operation of reading Status Code / User Code is similar to the operation of reading ID Code, just replace the corresponding instructions.

Write Enable (0x1500)

Before configuring SRAM (writing features), enter programming mode using "Write Enable (0x15)" instruction to receive the "Write Data (0x3B)" instructions.

Figure 5-32 Write Enable (0x15) Timing
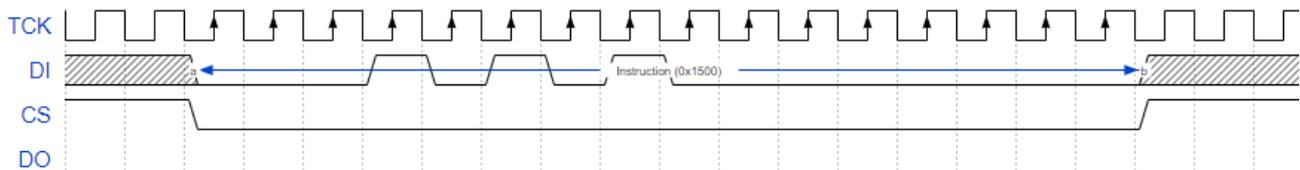
Note!

At CS high level, more than two clocks should be given to SCLK to drive FPGA to identify CS signal. This rule also applies to other instructions.

Write Disable (0x3A00)

After finishing sending data, exit programming mode using Write Disable. After exiting, the device can be awakened to enter the working state.

Figure 5-33 Write Disable(0x3A00) Timing
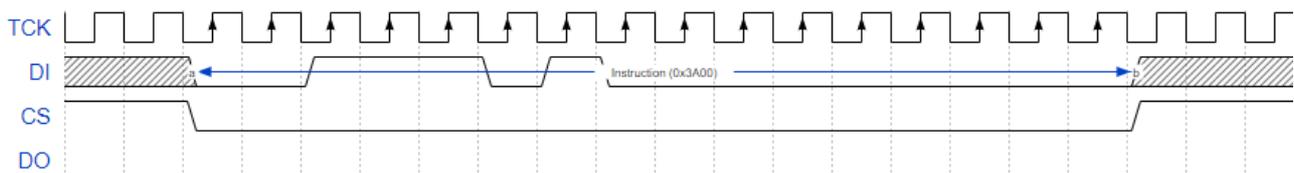
The timing of 0x1500 and 0x3A is basically the same. Instructions start at CS low level and the CS is pulled up after the instruction transmission is completed. Instructions following this timing are as follows: 0x3C00 (Reconfig / Reprogram), 0x1500(Write Enable), 0x3A000 (Write Disable), 0x1600(Program SPI Flash), 0x1200(Init Address), 0x0500(Erase SRAM).

In addition, SSPI is driven by an external clock, so if CS is at high before and after these instructions, more than two clocks are needed to enable FPGA to collect the state of CS.

Write Data (0x3B)

The fs file is sent directly to the FPGA device using the "Write Data (0x3B)" instruction.

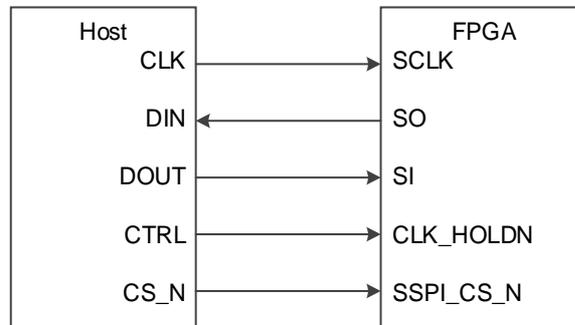Note that CS keeps low level in the process of data writing.

Figure 5-34 Write Data (0x3B) Timing
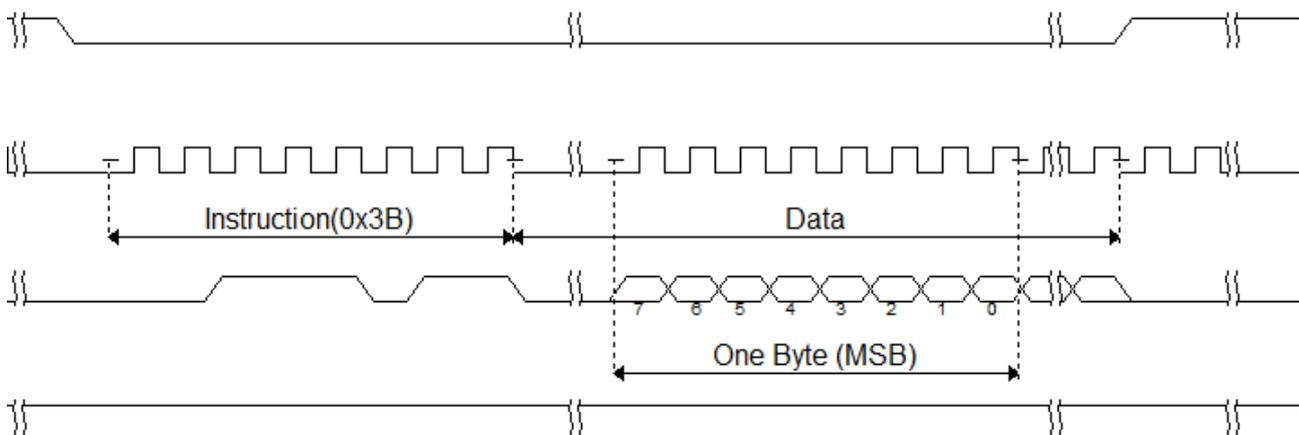
5.4.4 Connection Diagram for SSPI Configuration Mode

The connection diagram for configuring Gowin FPGA products via SSPI is shown in Figure 5-35.

**Figure 5-35 SSPI Configuration Mode Connection Diagram**

Note!

The figure above shows the minimum system diagram for the SSPI configuration. The value of the SSPI MODE is "001". The connection of the other fixed pins is shown in Figure 5-1.

In addition to SRAM, SSPI can be used to program external SPI Flash. The MODE value of the Flash programming is the same as the MODE value of SSPI configuration mode. Configuration data can be written to SRAM or an external Flash using Gowin programmer. The connection diagram for programming an external Flash via SSPI is shown in Figure 5-36.

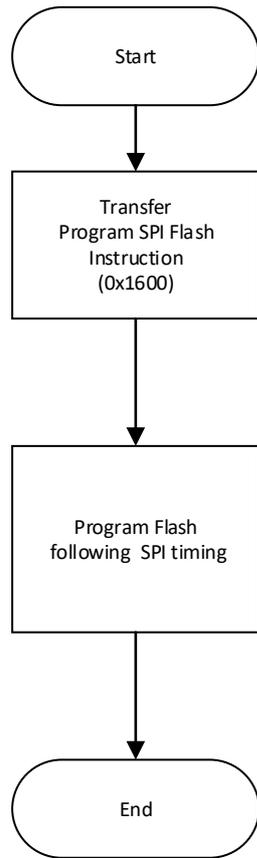**Figure 5-36 Connection Diagram of Programming External Flash via SSPI**
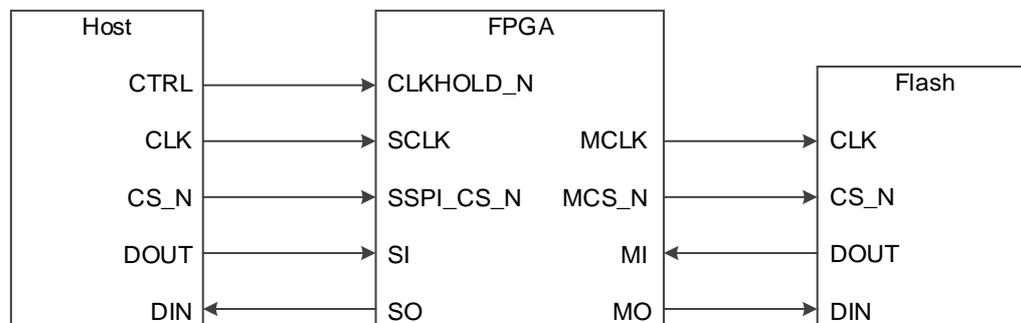
Note!

- All Arora family devices support programming external Flash via SSPI.
- For the LittleBee® family devices, currently only GW1N(R)-9 supports programming external Flash via SSPI.

Please refer to Figure 5-37 for the flow of programming external Flash via SSPI.

First, send the "Program SPI Flash" (0x1600) instruction to FPGA via SSPI. After this, the FPGA can forward SSPI to Flash, and the SSPI on the Host side can directly access Flash. Then, it can be programmed according to Flash timing.

Note that when reading data from Flash, the data being read back is delayed by one bit. For example, when SSPI reads Flash's ID Code, it needs to send an extra Clock to get the last bit.

Figure 5-37 The Flow of Programming External Flash via SSPI

```
                    ┌─────────────────┐
                    │      Start      │
                    └────────┬────────┘
                             │
                             ▼
                    ┌─────────────────┐
                    │    Transfer     │
                    │ Program SPI Flash│
                    │   Instruction   │
                    │    (0x1600)     │
                    └────────┬────────┘
                             │
                             ▼
                    ┌─────────────────┐
                    │                 │
                    │  Program Flash  │
                    │ following  SPI timing│
                    │                 │
                    └────────┬────────┘
                             │
                             ▼
                    ┌─────────────────┐
                    │       End       │
                    └─────────────────┘
```

5.4.5 Multiple FPGA Connection View in SSPI Mode

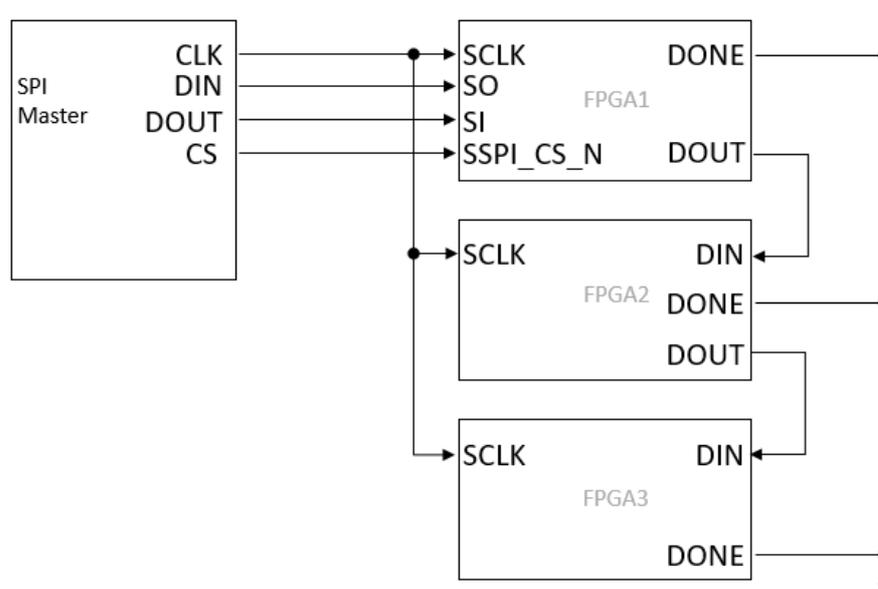**Figure 5-38 Multiple FPGA Connection Diagram 1**
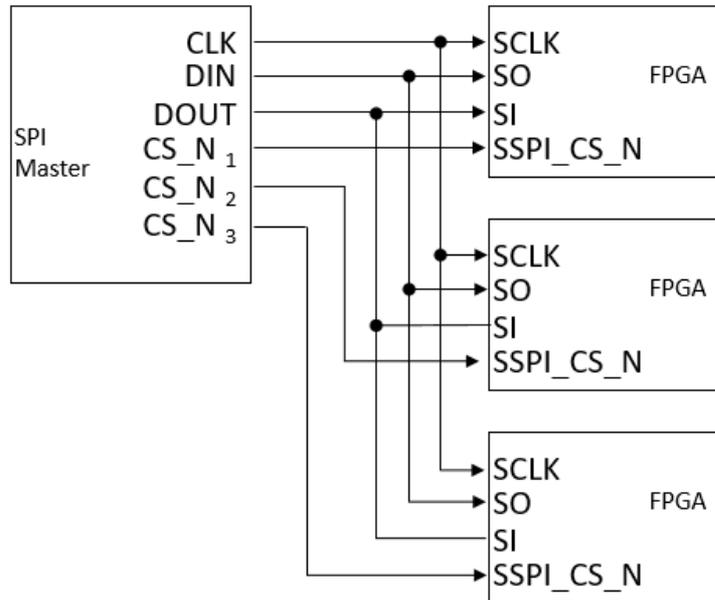
Figure 5-39 Multiple FPGA Connection Diagram 2

5.5 MSPI

In MSPI (Master SPI) mode, FPGA is as a Master and reads bitstream data from the external Flash via SPI port to complete configuration.

MSPI Configuration Process: Set the MODE pin to MSPI status, power on again or trigger RECONFIG_N at one low-level pulse, and the device will read bitstream data from the external Flash and complete configuration automatically.

According to the MSPI configuration features, remote upgrade requirements can be implemented: After starting the FPGA, if an upgrade

is required, users can remotely write the configuration data into the external Flash, and trigger RECONFIG_N or power up again to upgrade the system if the upgrade conditions are met.

MSPI Mode Pins

The configuration of the MSPI mode is shown in Table 5-15.

Table 5-15 Pin Description in JTAG Configuration Mode

| Pin Name	I/O	Description
RECONFIG_N	I, Internal weak pull-up	Low level pulse: Start GowinCONFIG
READY	I/O	High-level pulse: The device can be programmed and configured; Low level: Programming configuration for device is prohibited
DONE	I/O	High-level pulse: Successfully programmed and configured; Low-level pulse: Programming and configuration uncompleted or failed.
MODE[2:0]	I, Internal weak pull-up	MODE select signal, READY rising edge sample
MCLK	O	FPGA output clock
MCS_N	O	Chip selection signal, active low.
MO	O	FPGA outputs data to Slave
MI	I	Input data to FPGA through Slave
FASTRD_N	I	READY signal rising edge sampling High level: Read SPI mode (SPI instruction:0x03) Low level: Fast Read SPI mode (SPI instruction:0x0B)

Note!

The MSPI configuration mode clock frequency should not be greater than 70MHz. The Flash high-speed access mode and external pull-down FASTRD_N pin are required when the clock frequency is greater than 30MHz and less than 70 MHz. Leave the FASTRD_N pin floating if the clock frequency is less than 30 MHz.

Connection Diagram for MSPI Configuration Mode

The connection diagram for configuring Gowin FPGA products through MSPI is shown in Figure 5-40.

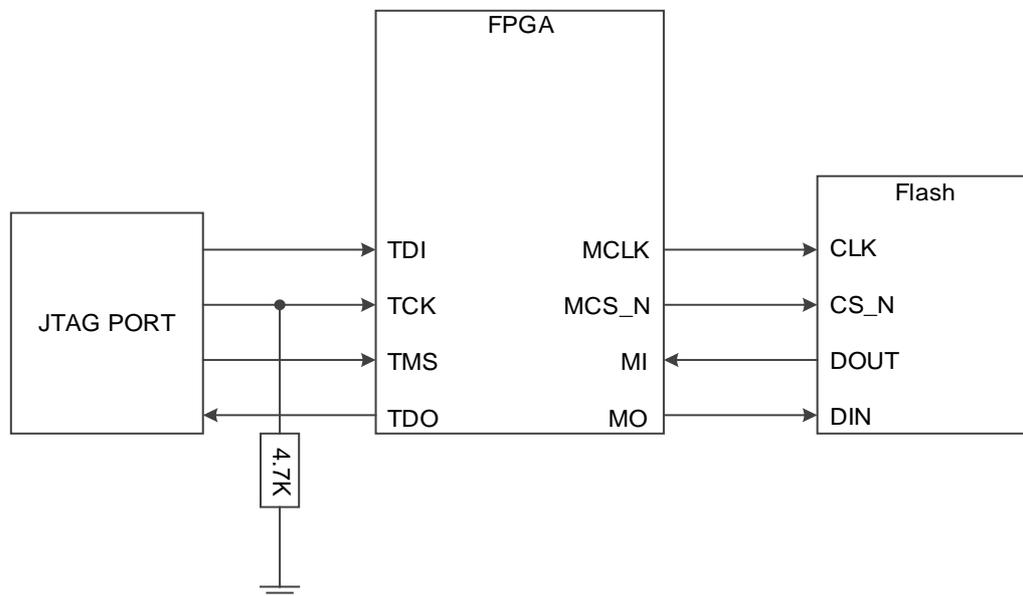**Figure 5-40 Connection Diagram for MSPI Configuration Mode**
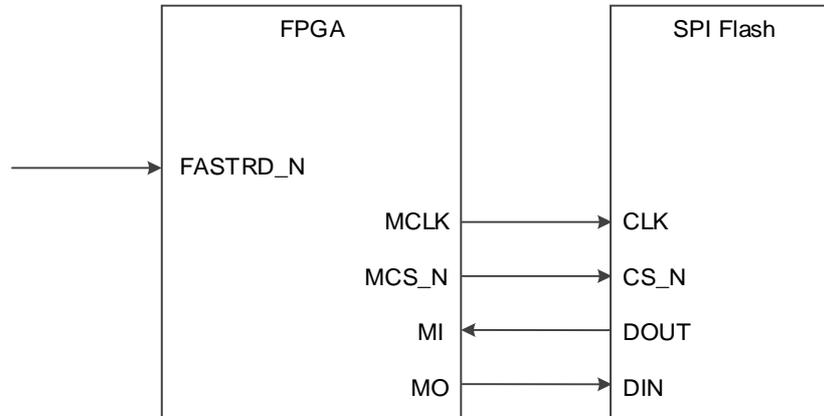
Note!

The figure above shows the minimum system diagram for the MSPI MODE. The value of the MSPI MODE is "010" (GW1N(R)) and "000" (GW2A(R)). The other fixed pins are shown in Figure 5-1. The FASTRD_N pin can remain floating in MSPI mode if the clock frequency is less than 30 MHz.

The connection diagram for programming data to external Flash is shown in Figure 5-41. The connection diagram for programming external Flash via the SSPI interface is shown in Figure 5-36.

Figure 5-41 Connection Diagram of JTAG Programming External Flash

Note!

The figure above shows the minimum system diagram of programming external Flash via JTAG. The connection for the other fixed pins is shown in Figure 5-1 .

Gowin FPGA products usually only support one time automatic MSPI configuration after power up. The GW1N (R)-9, GW2A (R)-18, and GW1NS series products are improved: GW2A (R)-18 series FPGA support retrying configuration once; GW1N (R)-9 and GW1NS FPGA support retrying

configuration twice. When the MSPI fails to configure after power up, the device can be reconfigured automatically according to the retry times supported. The factors that can lead to a failed configuration include false ID validation, false CRC check, and false instruction. The user can specify the SPI Flash address for retrying configuration, and write it through the Gowin software interface. This feature greatly reduces the risk of configuration failure, and thereby, ensures higher reliability of the user design.

MULTI BOOT

The derivative concept of MULTI BOOT refers to the FPGA reading bitstream data from different addresses in one same external Flash. Currently, the Gowin Programmer software supports the ability to program multiple bitstream data to external Flash without erasure, and the initial programming address is 0. The loading address of the latter bitstream data is written in previous bitstream data and the configuration is completed by triggering RECONFIG_N to switch the data stream file under the condition that the device power is on. FPGA products that support MSPI all support this mode.

Refer to the following steps for MULTI BOOT:

1. Open "BitStream" in Gowin software. Input the start address for the next BitStream in the text box following "SPI Flash Address", as shown in Figure 5-42;
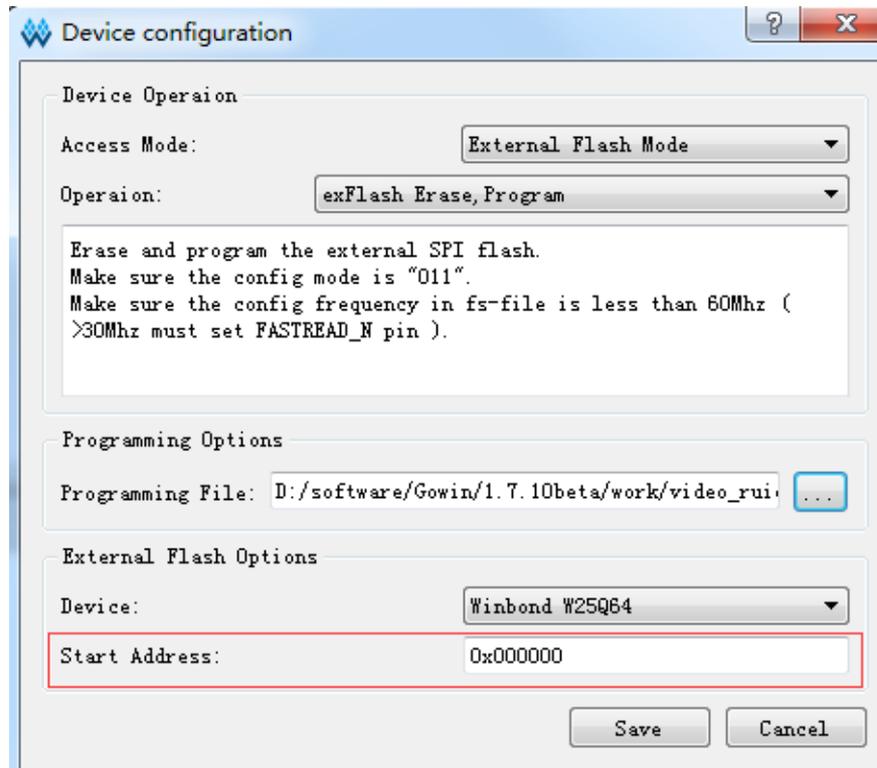
Figure 5-42 Input the Start address for the Next BitStream
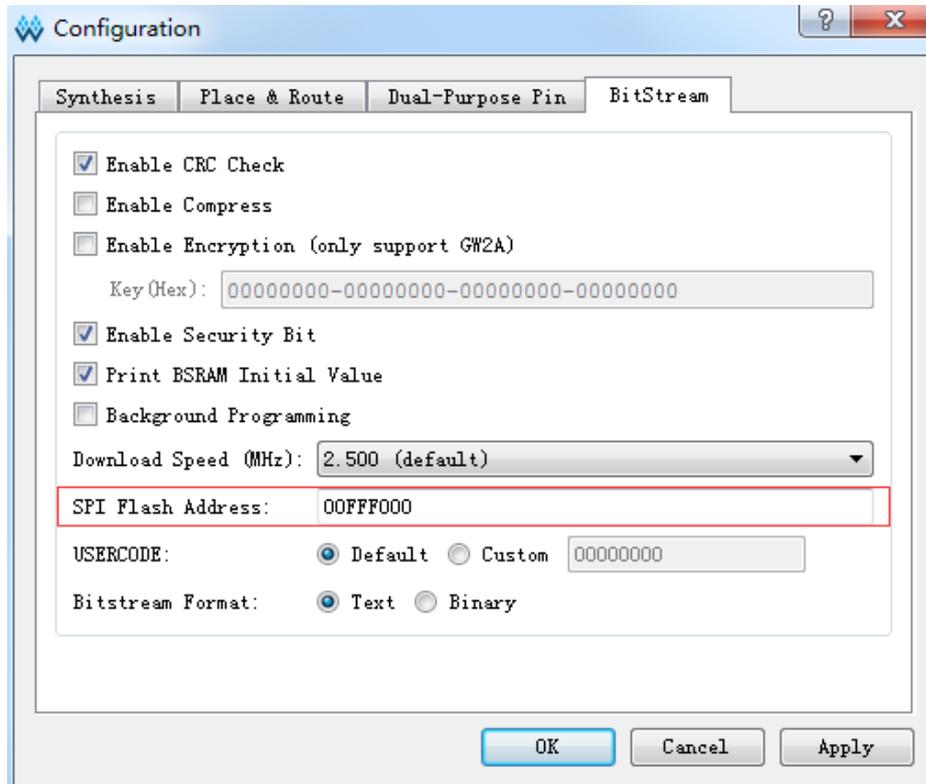
2. Select the external Flash mode in Programmer, set the start address of BitStream. This address should be the same as the start address set in

Step 1, as shown in Figure 5-43;

Figure 5-43 Set the Programming Address for the External Flash

3. Click "Save" to complete the setting of BitStream start address and programming address.
4. Trigger RECONFIG_N at one low pulse to realize the switching of multiple BitStreams.

Note!

●   MULTI BOOT needs to trigger RECONFIG_N to switch the configuration data during power on, and the start address is reset after power down.
●   You need to calculate the size of the bitstream data before using multiple configurations to ensure that the start address is not covered by the previous bitstream data;
●   The lower 12 bits of an SPI Flash start address is invalid and the address space of ADDR [23:12] can be set by users.

In addition to the introduction of configuring one FPGA via one Flash, Gowin FPGA products also support configuring multiple FPGAs with one Flash: The FPGA directly connected to the SPI Flash adopts MSPI mode, while the other FPGA devices use SSPI or SERIAL mode. For the specific operation, please refer to the following version. The connection diagram is shown in Figure 5-44.

Note!

Before configuring, set the MODE value of the FPGA to MSPI and SERIAL or MSPI and SSPI. Gowin FPGA products do not support the configuration of one FPGA with multiple Flashes.

Figure 5-44 Connection Diagram for Configuring Multiple FPGAs via Single Flash
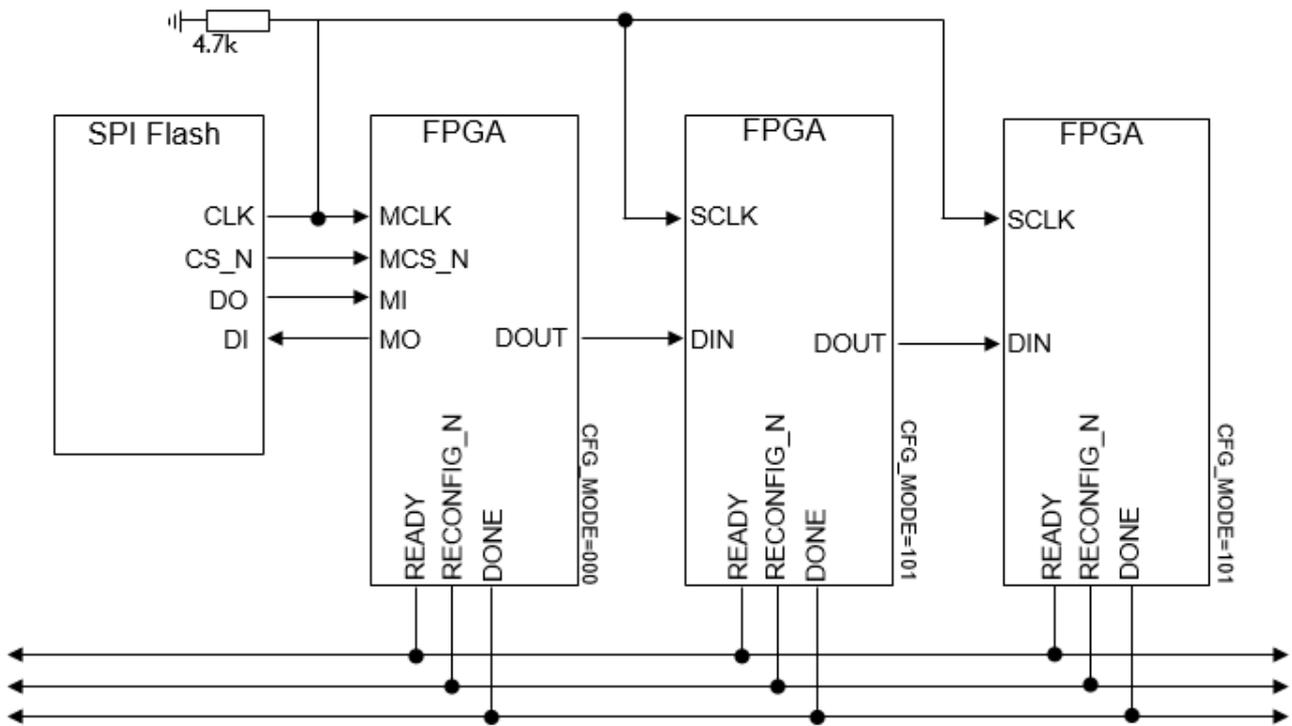
MSPI Configuration Timing

MSPI Download Timing is as shown in Figure 5-45.
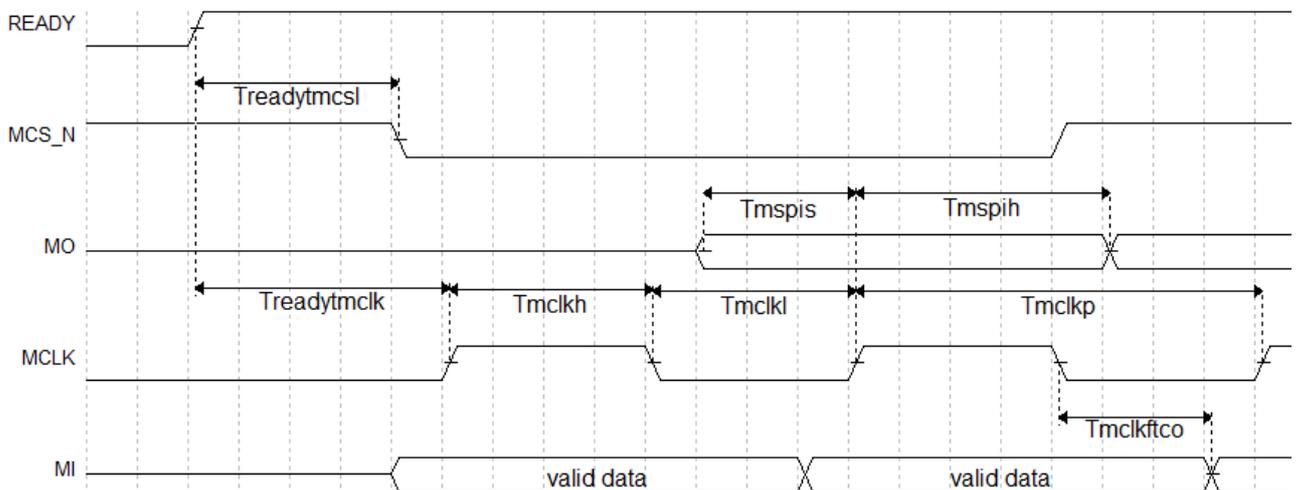
Figure 5-45 MSPI Download Timing

Table 5-16 shows the timing parameters.

Table 5-16 MSPI Configuration Timing Parameters

Name	Description	Min.	Max.
T_{mclkp}	MCLK clock period	15ns	-
T_{mclkh}	MCLK clock high time	7.5ns	-
T_{mclkl}	MCLK clock low time	7.5ns	-
T_{mspis}	MSPI PORT setup time	5ns	-
T_{mspih}	MSPI PORT hold time	1ns	-
$T_{mclkftco}$	Time from MCLK falling edge to output	-	10ns
$T_{readytmcsl}$	Time from READY rising edge to MCS_N low	100ns	200ns
$T_{readytmclk}$	Time from READY rising edge to first MCLK edge	2.8µs	4.4µs

Other than the power requirements, the following conditions need to be met to use the MSPI configuration mode:

- MSPI port enable
  RECONFIG_N is not set as a GPIO during the first configuration after power up or the previous programming.
- Initiate new configuration
  Power-on again or trigger RECONFIG_N at one low pulse.

Figure 5-46 Multiple FPGA Connection Diagram in MSPI Configuration Mode
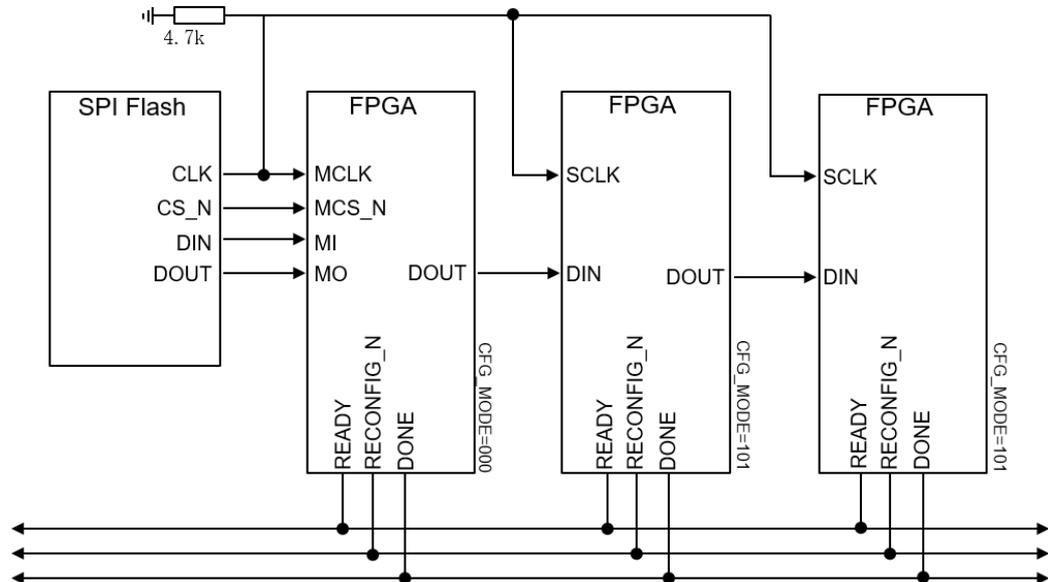
5.6 DUAL BOOT Configuration (Supported by LittleBee®
Family Only)

The DUAL BOOT mode is a configuration mode supported by the nonvolatile LittleBee® Family of FPGA products. In DUAL BOOT mode, FPGA first reads bitstream data from external Flash to complete configuration.
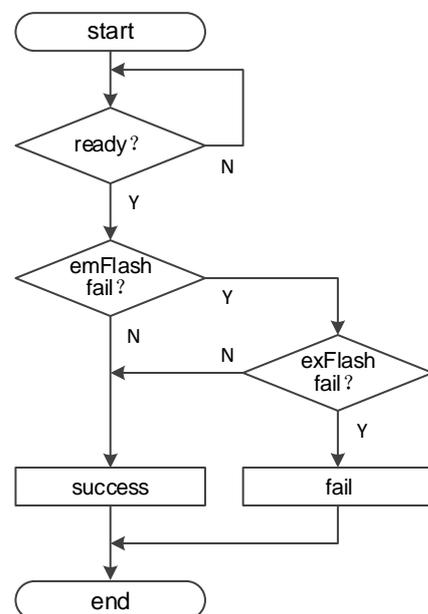
Note!

In DUAL BOOT mode, when the external Flash is empty or non-existent, FPGA will try to read data from the built-in Flash.

The specific MODE value needs to be selected for the DUAL BOOT MODE. No external connection is required for the built-in Flash. The connection diagram for reading from external Flash is the same as that of the MSPI mode. Please refer to Figure 5-40. In Dual BOOT mode, users can select where to save the configuration data required.

The Dual BOOT mode supported by GW1NS-2/GW1NS-2C device is slightly different from that of the other LittleBee® family devices. GW1NS-2/GW1NS-2C has double built-in Flash, so GW1NS-2/GW1NS-2C switches between the two built-in Flash in Dual BOOT mode.

The Dual Boot mode configuration flow is shown in Figure 5-47.

Figure 5-47 Dual Boot Flow Chart

Note!

When the MODE value is set to "110", the FPGA first attempts to configure from the external Flash.

GW1N(R)-9 and GW1NS series products support four times configuration in all DUAL BOOT modes.

- Start from the preferred storage path and attempt three times; if all attempts fail, start from the other storage path. The embedded Flash can only be started at "0" address;
- When the MODE value is "110", different startup addresses can be selected for the three attempts to start from external Flash. The startup address needs to be written to the bitstream through Gowin software in advance. If the configuration fails three times, the devices attempt to start from the built-in Flash.
- The GW1NS series of FPGA products support multiple restarts after failures, but the start address cannot be modified.

Note!

The lower 12 bits of an SPI Flash startup address is invalid and the address space of ADDR [23:12] can be set by users.

GW1N (R)-4 devices do not currently support automatic DUALBOOT configuration. Gowin provides users with DUAL BOOT configuration solution for these two devices. Please refer to TN101-1.0E_GW1N-4 FPGA Download DUAL BOOT Program for more details.
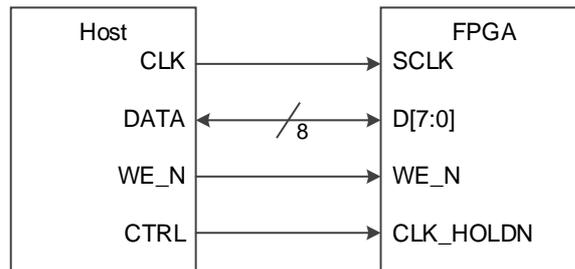
5.7 CPU Mode

In CPU mode, the Host configures Gowin FPGA products through the 8-bit data bus interface. CPU mode pins are shown in Table 5-17.

Table 5-17 CPU Mode Pins

Pin Name	I/O	Description
RECONFIG_N	I, internal weak pull-up	Low level pulse: Start GowinCONFIG
READY	I/O	High-level pulse: The device can be programmed and configured; Low level: Programming configuration for device is prohibited
DONE	I/O	High-level: Successfully programmed and configured; Low-level: Programming and configuration uncompleted or failed.
MODE[2:0]	I, internal weak pull-up	Configuration mode selection, READY rising edge sampling
SCLK	I	Input clock
CLKHOLD_N	I, internal weak pull-up	High: CPU operation is valid Low: CPU operation is invalid
WE_N	I	Read-write enable 0：Write 1：Read
D[7:0]	I/O	Data I/O port: Used as input pin in CPU mode, and used as output pin after configuration for verification

The connection diagram for the CPU mode is shown in Figure 5-48.

Figure 5-48 Connection Diagram for CPU Mode

Note!

The figure above shows the minimum system diagram of the CPU MODE. The MODE value is set to "111". The connections for the other fixed pins are shown in Figure 5-1.

Other than the power requirements, the following conditions need to be met to use the CPU configuration mode:

- CPU port enable
  RECONFIG_N is not set as a GPIO during the first configuration after power up or the previous programming.
- Initiate new configuration
  Power-on again or trigger RECONFIG_N at one low pulse.

5.7.1 Configuration Timing

Before configuration, make sure that MODE[2: 0]=111, and DONE will be pulled up after configuration. If DONE or READY is pulled down, the configuration fails.

In the configuration process, data bus D[7:0] is the MSB mode, and the FPGA reads the data at the SCLK rising edge.

Figure5-49 CPU Mode Configuration Timing
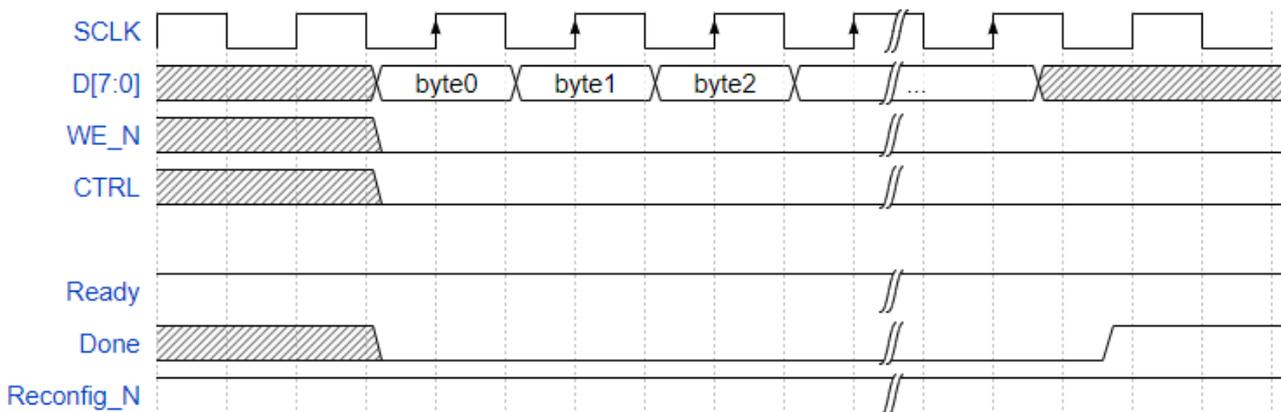
5.8 SERIAL Mode

In SERIAL mode, Host configures Gowin FPGA products via serial interface. SERIAL is one of the configuration modes that use the least number of pins. The SERIAL mode can only write bitstream data to FPGA and cannot readback data from FPGA devices; as such, the SERIAL mode cannot read information on the ID CODE and USER CODE and status

register. A definition of the pins employed in the SERIAL mode is provided in Table 5-18.

Table 5-18 Pin Definition in SERIAL Configuration Mode

Pin Name	I/O	Description
RECONFIG_N	I, internal weak pull-up	Low level pulse: Start GowinCONFIG
READY	I/O	High-level pulse: The device can be programmed and configured; Low level: Programming configuration for device is prohibited
DONE	I/O	High-level: Successfully programmed and configured; Low-level: Programming and configuration uncompleted or failed.
MODE[2:0]	I, internal weak pull-up	Configuration mode selection, READY rising edge sampling
SCLK	I	Input clock
DIN	I, internal weak pull-up	Input data
DOUT	O	Output data, only used in SERIAL configuration mode when FPGA cascading.

The connection diagram for the SERIAL mode is shown in Figure 5-50.

Figure 5-50 Connection Diagram for SERIAL Mode
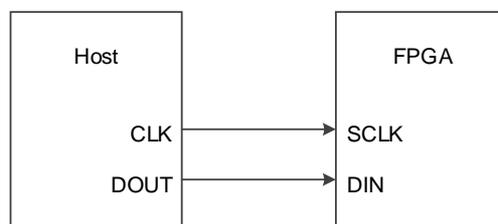
Note!

The figure above shows the minimum system diagram of the SERIAL MODE. The MODE value is set to "101". The connection for the other fixed pins is shown in Figure 5-1.

SERIAL Configuration Timing

See Figure 5-51 for the timing of SERIAL mode.
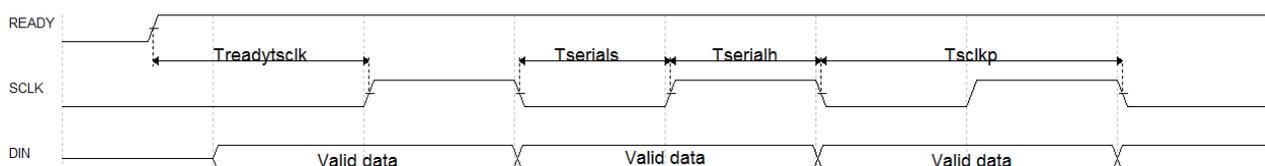
Figure 5-51 SERIAL Configuration Timing

Table 5-19 shows the timing parameters.

Table 5-19 SERIAL Configuration Timing Parameters

Name	Description	Min.	Max.
T_{sclkp}	SCLK clock period	15ns	-
$T_{serials}$	SERIAL PORT setup time	2ns	-
$T_{serialh}$	SERIAL PORT hold time	0ns	-
$T_{readytsclk}$	Time from READY rising edge to first SCLK edge	TBD	-

Other than the power requirements, the following conditions need to be met to use the SERIAL configuration mode:

- SERIAL port enable
  RECONFIG_N is not set as a GPIO during the first configuration after power up or the previous programming.
- Initiate new configuration
 Power-on again or trigger RECONFIG_N at one low pulse.

5.9 I²C Mode

In I^{2}C Mode, Gowin FPGA products are configured by Host via I^{2}C interface. I^{2}C Mode is one of the configuration modes that use the least number of pins. The I^{2}C mode can only write bitstream data to FPGA and cannot readback data from FPGA devices; as such, the I^{2}C mode cannot read information on the ID CODE, USER CODE, status register and read back check. A definition of the pins employed in the I^{2}C mode is provided in Table 5-20.

Table 5-20 Pin Definition in SERIAL Configuration Mode

Pin Name	I/O	Description
RECONFIG_N	I, internal weak pull-up	Low level pulse: Start GowinCONFIG
READY	I/O	High-level pulse: The device can be programmed and configured; Low level: Programming configuration for device is prohibited
DONE	I/O	High-level: Successfully programmed and configured; Low-level: Programming and configuration uncompleted or failed.
MODE[2:0]	I, internal weak pull-up	Configuration mode selection, READY rising edge sampling
SCL	I	Input clock
SDA	I/O	Input data or output ACK

The connection diagram for the I^{2}C mode is shown in Figure 5-52.

Figure 5-52 Connection Diagram for I²C Mode
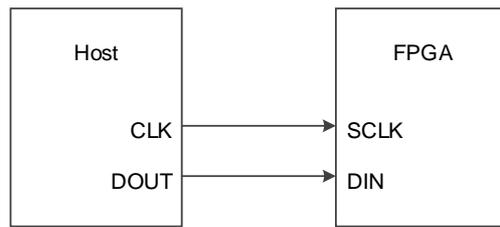
Note!

The figure above shows the minimum system diagram of the I²C MODE. The MODE value is set to "100". The connection for the other fixed pins is shown in Figure 5-1.

Figure 5-53 I²C Mode Timing
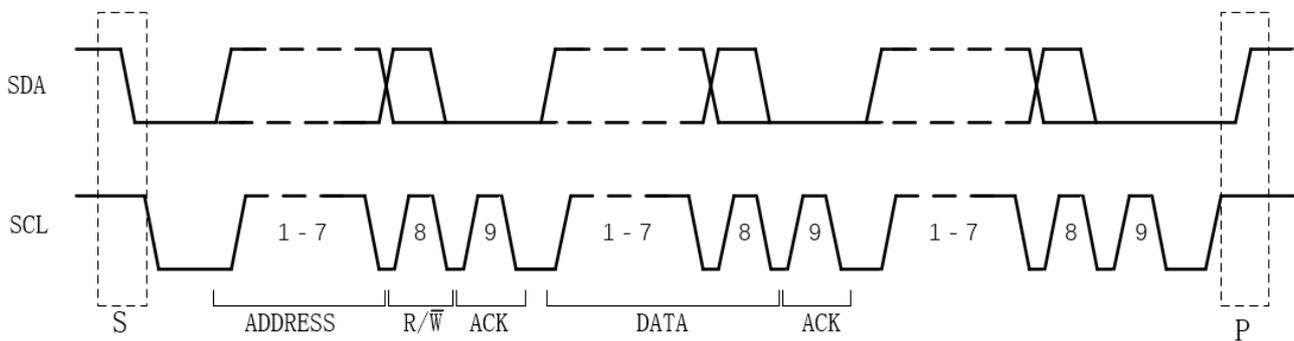
I²C is a serial transmission bus, which transmits data according to the protocol shown in the figure above. Under normal status, both SDA and SCL are at high level.

Table 5-21 I²C Configuration Timing Parameters

Prameter	Description	
S	Start condition	SCL is high level and SDA switches from high to low level.
P	Stop condition	SCL is high level and SDA jumps from low to high level.
ADDRESS	Address frame	A unique 7-bit or 10-bit sequence for each slave device that identifies the slave device when the master device is about to communicate with it.
R/W	Read/Write bit	Determines whether the master sends data to the slave (0) or reads data from the slave (1).
ACK	ACK/NACK 位	Each frame in the message is followed by an ACK/NACK bit, and Gowin FPGA returns 0 if correct.
DATA	Data	A data has 8bits, and the most significant bit is sent first.

All DATA on the I2C bus is transmitted in 8-bit bytes. Each byte sent by the transmitter, it releases the DATA line during the clock pulse 9, and the receiver sends back an answer signal.The response signal is a valid response bit (ACK bit) if it is low, indicating that the receiver has successfully received the byte. The response signal is a non-acknowledgement bit (NACK) if it is high, which generally indicates that the receiver did not succeed in receiving the byte. The requirement for the ACK feedback is that the receiver pulls the SDA line low during the low level prior to the 9th clock pulse and ensures a stable low level during the high level of the clock.If the receiver is the master, after it receives the last byte, it sends a NACK signal to notify the controlled sender to end the data transmission and releases the SDA line for the master receiver to send a stop signal.

Each bit of data transmitted on the I2C bus has a corresponding clock pulse (or synchronous control), that is, each bit of data is transmitted serially on the SDA bit by bit based on the SCL serial clock. During data transfer, the level on the SDA must remain stable, with the low level being data 0 and the high level being data 1, while the SCL is high.The level on the SDA is allowed to change state only while the SCL is low.Logic 0 has a low voltage level and Logic 1 has a high voltage level.As shown in the figure below.

SDA stable;
Data valid
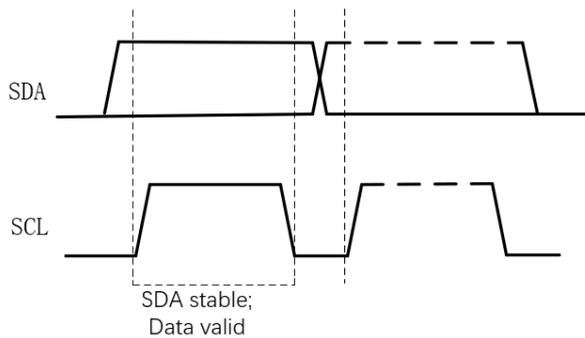
The list of I^{2}C mode supported by Gowin FPGA devices is as shown in the table below.

Mode	Device	Frequency	Address
SRAM	GW1N-2 （IDCode:0x0120681B）	100Khz~1.33Mhz	7'b1010_000
Embedded Flash	GW1N-2 （IDCode:0x0120681B）	1.33Mhz±1%	7'b1011_000
External Flash			

Note!

If you use I^{2}C to write Flash, the bitstream file needs to be conveted into specific bitstream file first. The conversion tool is included in Gowin Programmer, and the name after conversion is suffixed with ". I2C ".

Other than the power requirements, the following conditions need to be met to use the I^{2}C configuration mode:

- I^{2}C port enable
  RECONFIG_N is not set as a GPIO during the first configuration after power up or the previous programming.
- Initiate new configuration
  Power-on again or trigger RECONFIG_N at one low pulse.

6 Bitstream File Configuration

The features of Gowin FPGA products need to be configured and programmed using Gowin software. The settings mainly include configuration pins multiplexing options and bitstream data configuration options. This chapter describes the bitstream file configuration. For the details about the configuration pin reuse, please refer to 4.1.2 Configuration Pin Multiplexing.

To transfer the configuration data safely and accurately, the CRC calibration algorithm has been incorporated by default in the FPGA bitstream file, and the security bit is set. During the process of data configuration, input data is checked in real time. The wrong data cannot wake up the device, and the DONE signal is pulled down. After the configuration of the bitstream with security bit is complete, data readback cannot be performed.

6.1 Configuration Options

Please refer to Figure 6-1 for the related configuration data setting interface. The options include CRC enable, bit stream data compression, encryption key settings, security bit settings, MSPI configuration frequency selection, SPI Flash start address settings in multiple configuration modes, USER CODE setting, etc. The lower 12 bits of an SPI Flash startup address is invalid and the address space of ADDR [23:12] can be set by users.

Figure 6-1 Configuration Options
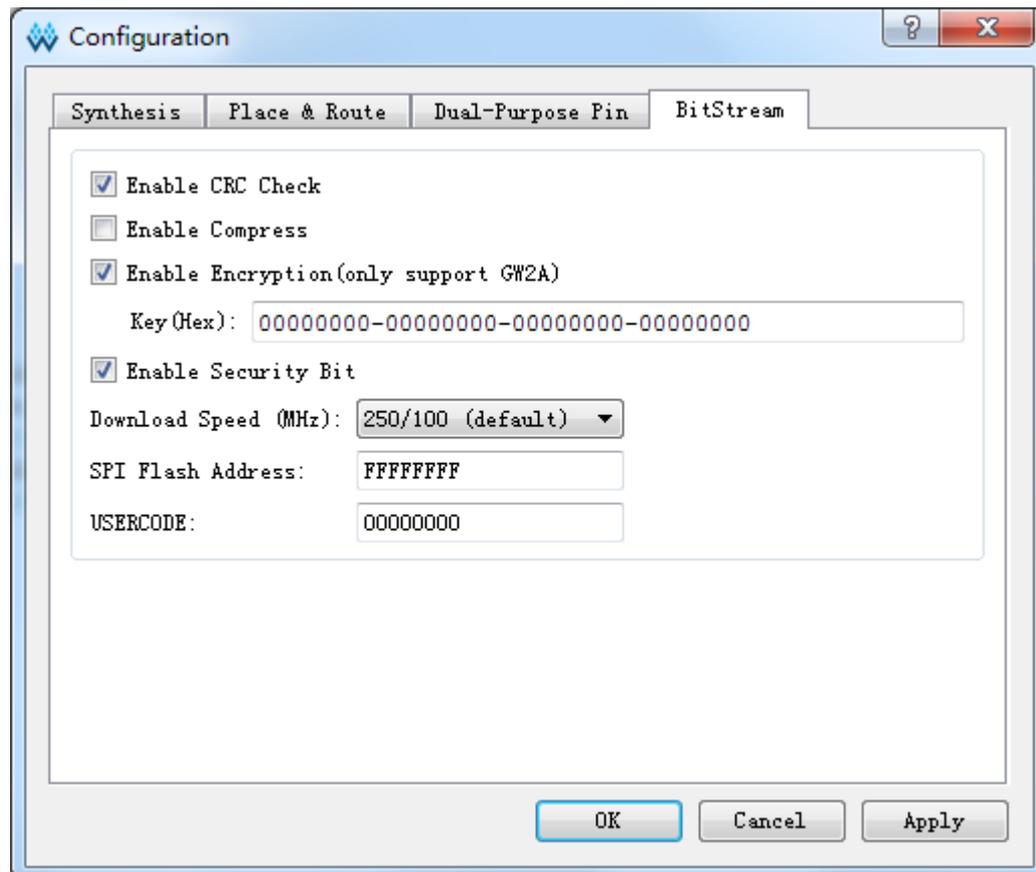
Note!

The security bit setting is forcibly checked after Gowin software verifies the encryption key setting option. In addition to ensuring the data is secure during the transmission process, using these bitstream settings during configuration also prevents any readback, thereby ensuring maximum protection of user data.

6.2 Configuration Data Encryption (Supported by Arora Family only)

The Gowin Arora® Family of FPGA products support bitstream data encryption, using the 128 AES encryption algorithm. Please refer to the following steps for the data encryption configuration:

1. Enter the encryption KEY (KEY) in Gowin software interface to generate the bitstream data;
2. Enter the decryption key in Gowin Programmer;
3. After encrypted bitstream data is loaded into the device, FPGA compares the data that has been loaded with the decrypt key values stored in advance.
   If data parsing succeeds, the device finishes configuration and begins to work; if data parsing fails, the device cannot work, and READY and DONE are pulled down.

6.2.1 Definition

- AES encryption key: AES private key used in AES encryption algorithm,

specified by users. Referred to as "key" in this manual.
- AES encryption key length: 128 bits;
- Key: An abbreviation for AES encryption key. GW2A(R) series of FPGA products offers an address with 128 bits length to store Key;
- Lock: To ensure the security of AES Key, it is used to control the read permissions for the Key. This operation is named as "lock" in this manual. When it's locked, all the read back data is 1.

6.2.2 Enter Encryption KEY

Refer to the steps below to write the encryption keys in Gowin software:
1. Open the corresponding project in Gowin software;
2. Select "Project > Configuration > Dual Purpose Pin" from the available menu options;
3. Click "BitStream", check "Enable Encryption (only support GW2A)" and input the key value, as shown in Figure 6-2.

Figure 6-2 Encryption Key Setting Method
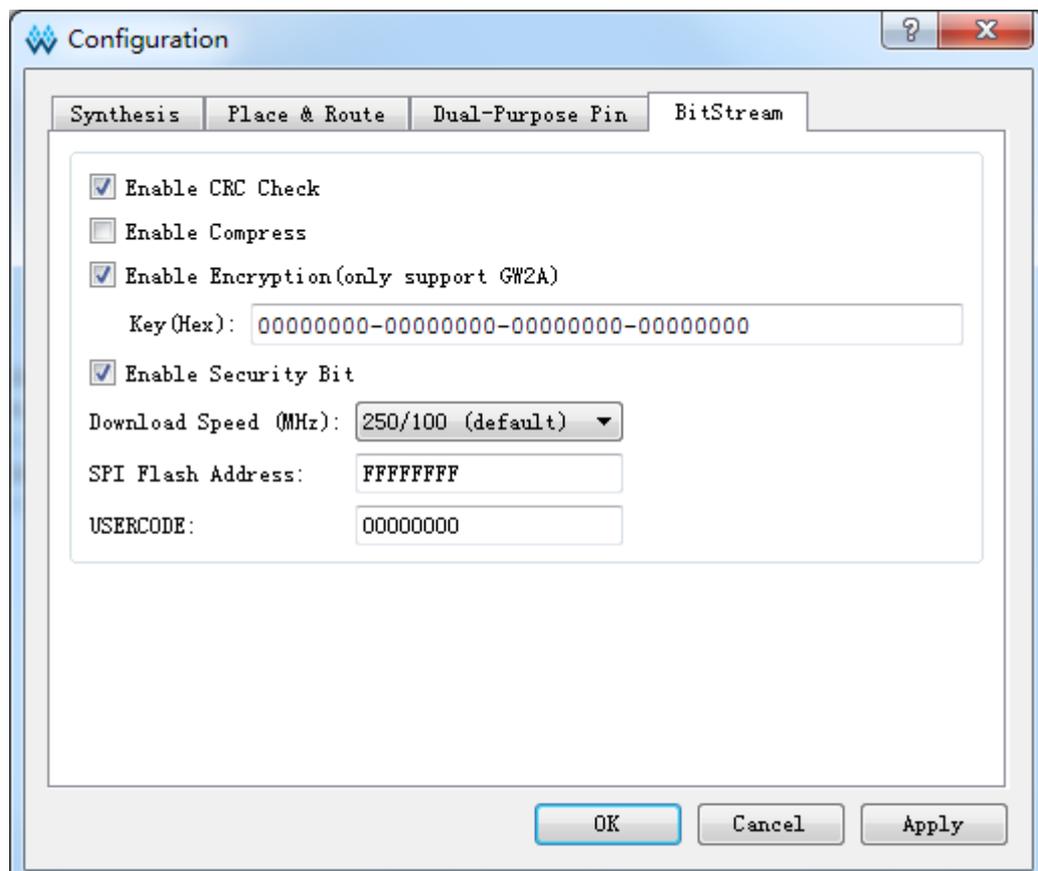
After setting the encryption key successfully, write the decrypted key to the FPGA key storage area for the device to analyze the encrypted bitstream data to complete the configuration.
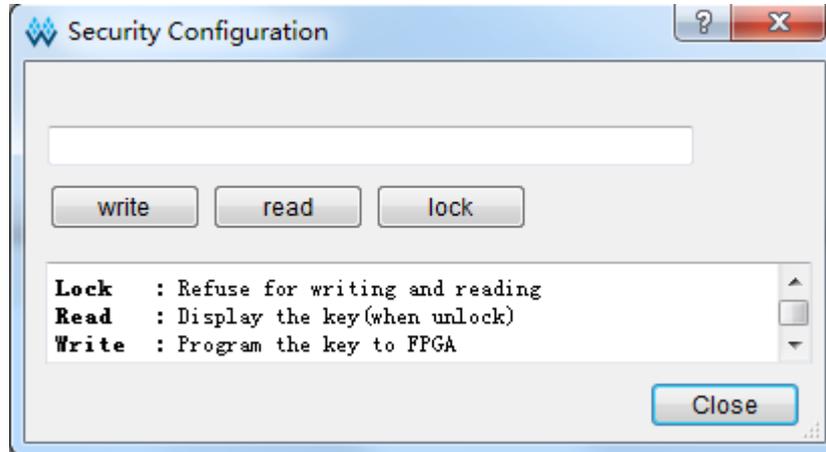
6.2.3 Enter the Decrypt Key

To input the decryption key, refer to the following steps:
1. Open the Gowin programming software;

2. Scan the FPGA device;
3. Right-click on the device name and select "Configure Security";
4. Enter the encrypted key value in the pop-up interface, click "write" and write the value to the FPGA, as shown in Figure 6-3.

Figure 6-3 Setting the Decryption Key

After the decryption key is written successfully, readback the written value via the "Read" button on the interface to verify.

After the key is written successfully, users also can select to "lock" it in FPGA via the Lock command. Once you have performed this action, any read and write key operations will be invalid, the key value cannot be modified, and all read bits are all "1".

After the decryption key is set, the encrypted bitstream data will only work when the data matches the decryption key. The key does not affect the non-encrypted bitstream data.

Note!

The initial value of the Gowin FPGA keys is 0. If a key value is changed to 1, it cannot be changed back to 0. For example, the key value written during an operation is 00000000-00000000-00000000-00000001, and the last bit of the modified key must be 1.

6.2.4 Programming Operation

Gowin Programmer offers the tool for programming AES encryption key. Open this tool by clicking "Tools > Security" in Gowin Software, as shown in Figure 6-4.

Figure 6-4 AES Security Configure
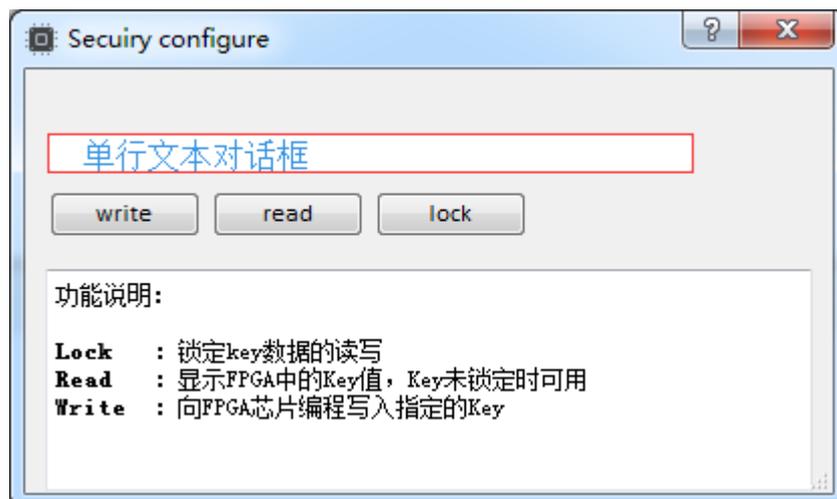
This configuration contains the following three parts:
● Write: Write Key;
● Read: Read Key;
● Lock: Lock read and write access to the Key.

Write

1. Write the user-defined Key to the text box in the figure above;
2. Click "Write" button;
3. Return the validation result after running.

Read

Click "Read" button to validate the written AES encryption key again. The Key that is read from the tool will be displayed in the text box in the figure above.

Lock

Click "Lock" to lock the read and write permission of Key. If it is locked, the Key cannot be read or written.

6.2.5 Programming Flow

Figure 5-21 shows the flow of how to program or lock the AES key. All the flows are based on JTAG protocol.

Check ID CODE

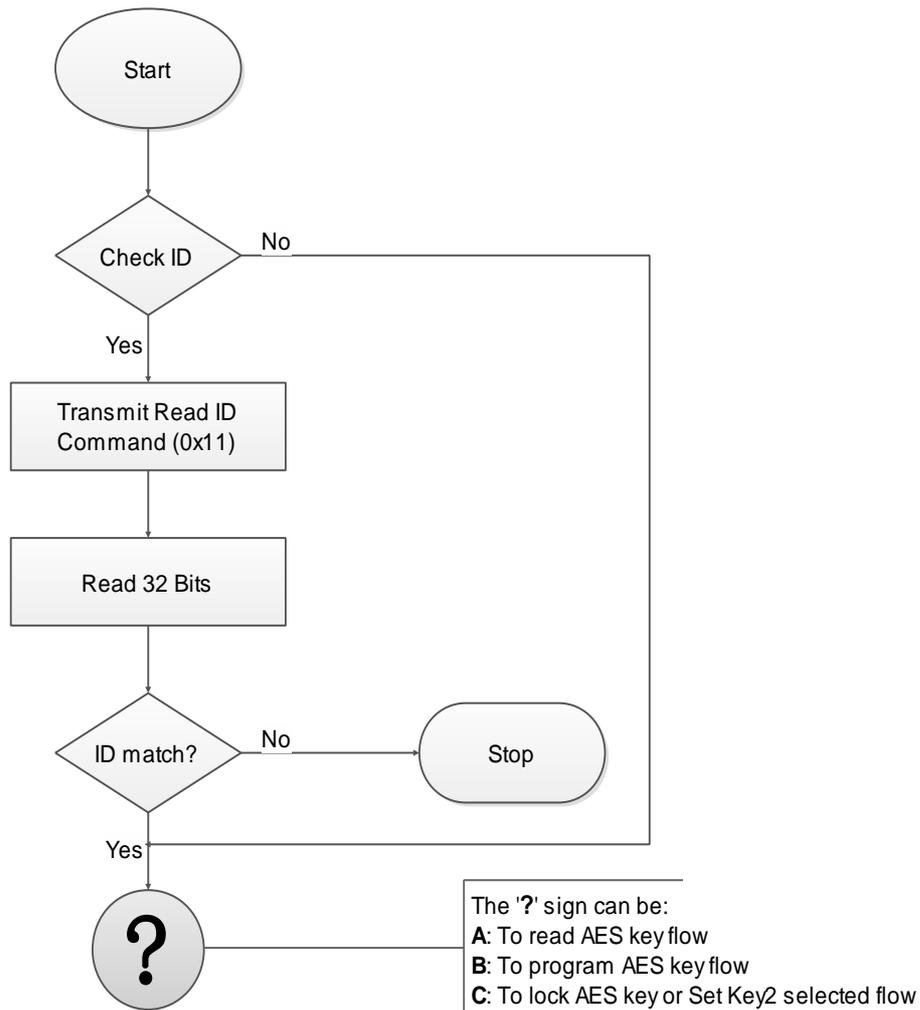Check the device ID to determine whether the JTAG protocol works properly and whether the programing object is correct to avoid misoperation.

Figure 6-5 Prepare

```
                    ┌─────────┐
                    │  Start  │
                    └────┬────┘
                         │
                         ▼
                    ╱ Check ID ╲ ──No──────────────────┐
                    ╲          ╱                        │
                         │ Yes                          │
                         ▼                              │
              ┌──────────────────┐                      │
              │ Transmit Read ID │                      │
              │ Command (0x11)   │                      │
              └────────┬─────────┘                      │
                       │                                │
                       ▼                                │
              ┌──────────────────┐                      │
              │   Read 32 Bits   │                      │
              └────────┬─────────┘                      │
                       │                                │
                       ▼                                │
                  ╱ ID match? ╲ ──No──→  ( Stop )       │
                  ╲           ╱                         │
                       │ Yes ←──────────────────────────┘
                       ▼
                     ( ? )
```

The '**?**' sign can be:
A: To read AES key flow
B: To program AES key flow
C: To lock AES key or Set Key2 selected flow

Read AES Key

Figure6-6 Read AES Key Flow

```
           ( A )
             |
             v
   +-------------------+
   | Transmit ISC Enable|
   | Command (0x15)    |
   +-------------------+
             |
             v
   +-------------------+
   | Transmit Read Key |
   | Command (0x25)    |
   +-------------------+
             |
             v
   +-------------------+
   |      Delay        |
   |     100 ms        |
   +-------------------+
             |
             v
   +-------------------+
   |   Read 128 Bits   |
   +-------------------+
             |
             v
   +-------------------+
   | Transmit ISC Disable|
   | Command (0x3A)    |
   +-------------------+
             |
             v
   (      Stop      )
```
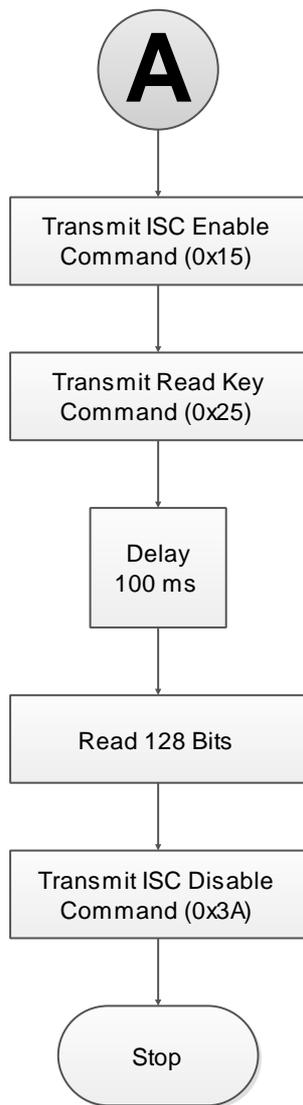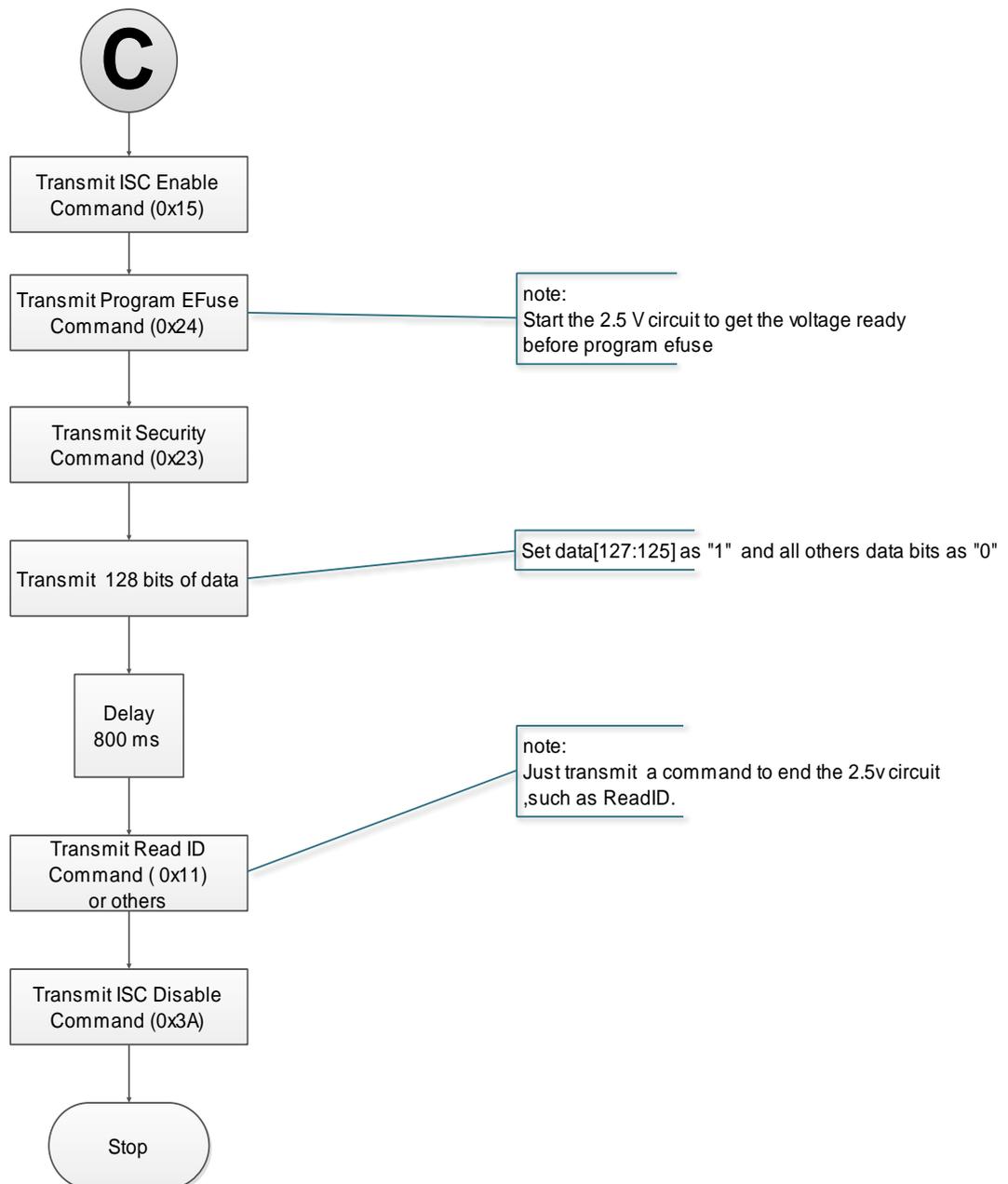
Program AES Key

Figure 6-7 Program AES Key Flow
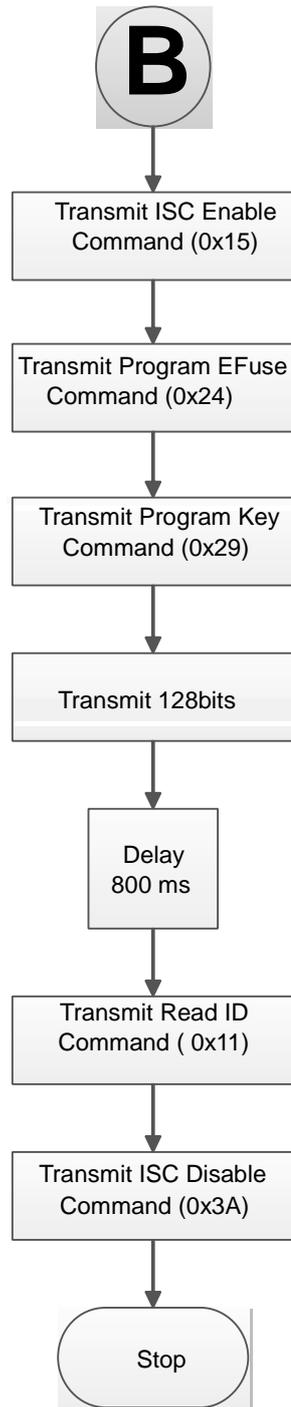
Lock AES Key

Locking the AES Key prevents the Key leakage. After locking the AES Key, you will not be able to read and configure the AES Key.

Figure 6-8 Lock AES Key Flow

6.3 Configuration File Size

The Gowin bitstream format can be Text (ASCii) with annotations or Binary with no annotations. The file with a .fs suffix is a text format file. Lines beginning with "//" are annotations. The others is the bitstream data. The file with a .bin suffix is a binary format file, with no annotations. This binary format file is commonly used for embedded programming. Users can configure the bitstream file format in Gowin software.
1. Open the Gowin software;

2. On the Process tab, right click Place & Route and then click "Configuration > Bitstream";

3. In the options of Bitstream Format, select Text or Binary, as shown in Figure 6-9.

Figure 6-9 Bitstream Format generation
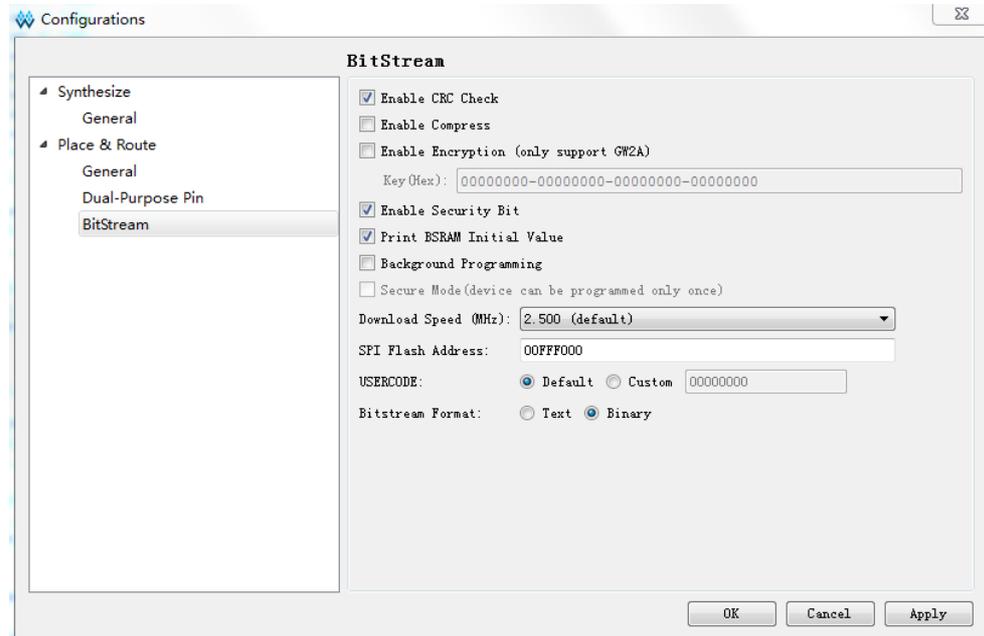
Gowin supports compressing bitstream data. The compression ratio is related to the user design. This manual only provides uncompressed configuration file sizes, as shown in Table 6-1.

**Table 6-1 Gowin FPGA Products Configuration File Size (Max.)**

LUT	Max. Configuration File Size
1,152	84 KBytes
4,608	217 KBytes
8,640	435 KBytes
20,736	887 KBytes
54,720	2269 KBytes

Note!

The data in the table is the file size in binary format, and the configuration file is not compressed. If SPI Flash is used to store bitstream file, memory margin is required.

6.4 Configuration File Loading Time

Gowin FPGA can be used as Master to read bitstream files from Flash and configure SRAM, including Autoboot mode and MSPI mode. In Autoboot mode, FPGA reads bitstream files from internal Flash. In MSPI mode, FPGA reads bitstream files from external Flash. When the FPGA is powered on and ready, it starts to read bitstream files, and when the loading is done, the FPGA enters the User Logic state, as shown in the figure below.
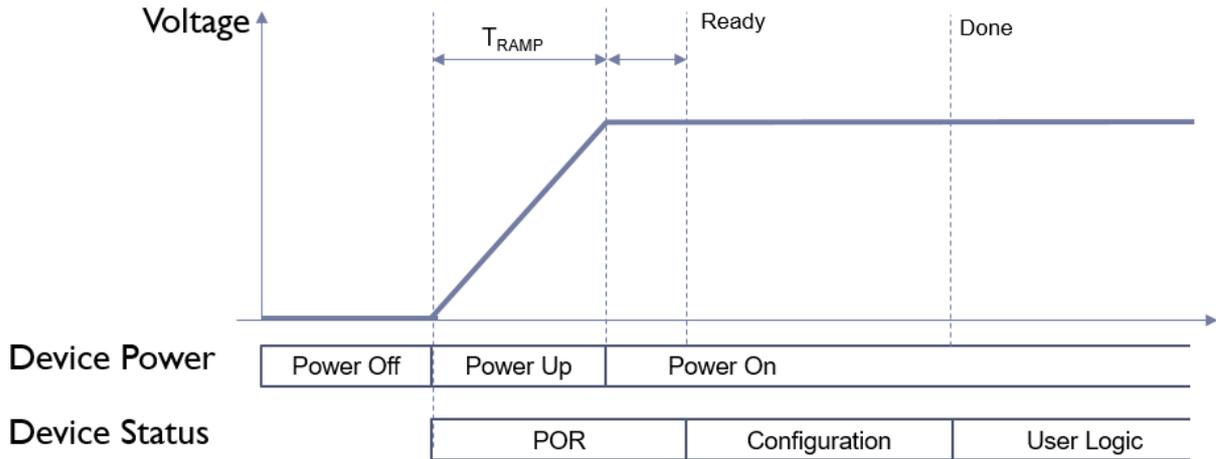
Both LittleBee® family and Arora family of GOWINSEMI FPGA devices support MSPI mode, that is, after the device is powered on, it can read bitstream files from the external SPI Flash and then complete the configuration. The default frequency of reading configuration file is 2.5 MHz. One bit is read at each SPI clock, so the required loading time can be calculated according to the file size. The clock frequency of reading SPI Flash in MSPI mode can be up to 125 MHz. Note that the FastRead_n pin should be grounded at the same time when Fast Read SPI （0x0B）is used.

The LittleBee® family devices support not only MSPI mode, but also Autoboot mode. The loading frequency is 2.5 MHz by default, and Autoboot mode loads one byte (8 bits) per clock. The loading time varies depending on the configuration file size, load frequency, and per-clock loading width. Due to the different process of the embedded Flash, the maximum Autoboot loading frequency for different devices is also different. The specific maximum loading speed is as shown in Table 6-2 below.

Table 6-2 Loading Frequency of Config File

Device	Max. Loading Frequency of Autoboot	Max. Loading Frequency of MSPI
GW2A-55/55C	–	125 MHz
GW2A-18/18C		
GW2AR-18/18C		
GW2ANR-18C		
GW1N-1	26 MHz	
GW1N-1S		
GW1NS-2	33 MHz	120 MHz
GW1NSR-2		
GW1NS-2C		
GW1NSR-2C		
GW1NSE-2C		
GW1NZ-1	40 MHz	
GW1N-2		

Device	Max. Loading Frequency of Autoboot	Max. Loading Frequency of MSPI
GW1N-2B		
GW1NSER-4C		
GW1NS-4		
GW1NSR-4		
GW1NS-4C		
GW1NSR-4C		
GW1N-4B		
GW1NR-4B		
GW1NRF-4B		
GW1N-4		
GW1NR-4		
GW1N-6		
GW1N-9		
GW1N-9C		
GW1NR-9		
GW1NR-9C		

The bitstream file loading time in MSPI mode is as shown in Table 6-3.

Table 6-3 Loading Time in MSPI Mode

Number of LUT4	Max. Configuration File	Loading Time (ms, when Frequency =2.5 M z)	Loading Time (ms, when Frequency =25 MHz)	Loading Time (ms, when Frequency =41.6 MHz)	Loading Time (ms, when Frequency =62.5 MHz)
1,152	84 KBytes	275	28	17	11
4,608	217 KBytes	711	71	42	28
8,640	435 KBytes	1425	142	85	57
20,736	887 KBytes	2906	290	174	116
54,720	2269 KBytes	7435	743	446	297

The bitstream file loading time in Autoboot mode is as shown in Table 6-4.

Table 6-4 Loading Time in Autoboot Mode

Number of LUT4	Max. Configuration File	Loading Time (ms, when frequency =2.5 MHz, default frequency)	Loading Time (ms, when Frequency =25 MHz)	Loading Time (ms, when Frequency =31.25 MHz)
1,152	84 KBytes	34	4	3
4,608	217 KBytes	88	9	7
8,640	435 KBytes	178	17	14

What is listed above is the reference of loading time. From power on to configuration completion of the device, in addition to the configuration time, there are also the power on time (Tramp) and initialization time of the device. The specific power on time is related to the power supply device. Therefore, the approximate time of FPGA from power on to loading completion can be calculated according to the following formula:

Autoboot mode:

$T_{loading\ time}$ = POR time + Number of Data Stream Bits /8/ Clock Cycle

MSPI mode:

$T_{loading\ time}$ = POR time + Number of Data Stream Bits /clock cycle

$T_{loading\ time}$ = POR time + Number of Data Stream Bits /clock cycle

7 Safety Precautions

Security is a key factor for users to design FPGA. Combined with GOWINSEMI devices features, Gowin programmer offers a series of safety precautions, which provides a perfect security mechanism for users' bitstream data.

Safety precautions consist of three stages:

- Before configuration, Gowin programmer checks the validity of the bitstream;
- During configuration, GOWINSEMI device verifies the accuracy of the transmission data in real time;
- After configuration, GOWINSEMI device enters the working state, masking any readback requests.

The details of the three stages are as follows:

Before Configuration

Gowin programmer can be used to configure Gowin FPGA by following the steps outlined below.

1. Connect the device that needs to be configured;
2. Start Gowin programmer to start scanning, and the connected FPGA devices can be identified automatically;
3. Select the bitstream and configuration mode to configure the device.

During the process outlined above, Gowin programmer will read the connected device ID first, and then compare this with the bitstream ID that users selected. The configuration can only proceed when the two IDs are identical, or the bitstream selected by users will be regarded as illegal data, resulting in configuration failure.

Note!

GOWINSEMI products have specific IDs that distinguish them from the other series of products. The bitstream generated by Gowin Software contains an ID verification directive, as such, users only need to select the specific device when creating a new project.

During Configuration

The device reads and verifies the bit stream ID first, and configuration starts if verification passes. To prevent bitstream modifications or possible transmission errors, GOWINSEMI devices adopt CRC to ensure bitstream is written in correctly. The specific process is outlined below.

Following each address segment of the bitstream generated by Gowin software, CRC is added. GOWINSEMI devices generate CRC in the process of receiving data and compares them with the check codes received. If a CRC error is detected, any data transmitted following this error will be ignored. The "DONE" indicator will not light up after configuration, and the CRC error message will be displayed on the Gowin programmer interface.

After Configuration

After configuration, the device bitstream will be loaded to the SRAM or on-chip Flash according to the configuration mode selected. (On-chip Flash is supported by the LittleBee® Family of FPGA products only.)

- If the data is loaded to the SRAM, Gowin software will set the security bit automatically in the process of bitstream generation, and no user can read SRAMs.
- If the data is loaded to the on-chip Flash, the Flash will be configured as the AUTO BOOT mode after Flash configuration is complete. Any reading requests will be prohibited.

The AUTO BOOT mode of the LittleBee® Family of FPGA products does not require external connections, so this greatly reduces the risk of data interception and provides the user with higher security. DUAL BOOT provides a selection for users with the option to write the configuration data to off-chip Flash as required.

Note!

GOWINSEMI takes no responsibility for the security of the off-chip Flash.

8 Boundary Scan

The boundary scan operation is an extension of the JTAG configuration mode. The scanning chains contain long chain and short chain. The long chain is mainly combined with BSDL file for device testing. The short chain is mainly used to erase and read and write the external Flash on the FPGA chain.

To perform a boundary scan, follow the steps outlined below:

1. Connect the FPGA development board to the PC and then power up;
2. Open Gowin programmer and scan the connected devices;
3. Double-click in the "Operation" field and select "External Flash Mode" and the related bscan operation, as shown in Figure 8-1.

Figure 8-1 Boundary Scan Operation Schematic Diagram
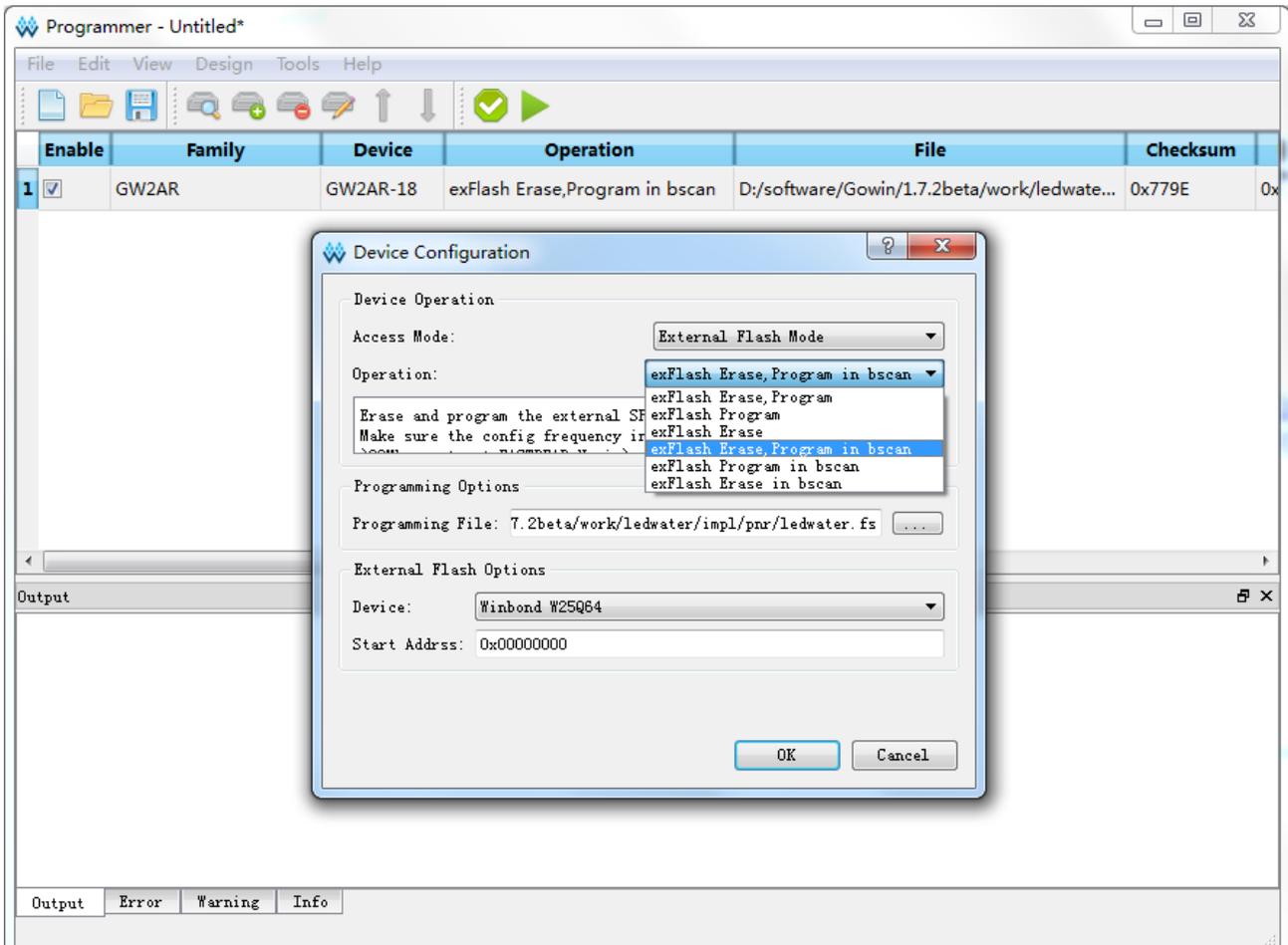
The boundary scan operation can only be performed on the external Flash of FPGA and cannot be used to program the embedded Flash or SRAM. This operation is irrelevant with the FPGA MODE value, but it is slower than that of the external Flash programming via JTAG.

9 SPI Flash Selection

The external SPI Flash device operation instructions supported by Gowin FPGA products are shown in Table 9-1. The Mxic and Winbond products are all in accordance with the requirements. In principle, if the read instruction and the quick read instruction are as shown in Table 9-1, Gowin FPGA can read data from this Flash.

Table 9-1 SPI Flash Operation Instruction

Operation	Instruction
Read	0x03
Fast_Read	0x0B

Note!

The Flash read instructions supported by Gowin FPGA must have at least one 03 or 0B. Use the regular reading instruction if the clock frequency is no higher than 30 MHz. Use the fast reading instruction if the clock frequency is higher than 30 MHz. Fast read requires the FASTRD_N pin to be pulled down, and the clock frequency cannot be higher than 70MHz.