Logos Family FPGAs Configuration User Guide

(UG020005, V2.3)

(13.07.2022)

Revisions History

Document Revisions

Version	Date of Release	Revisions
V2.3	13.07.2022	Initial release.

About this Manual

Terms and Abbreviations

Terms and Abbreviations	Meaning
JTAG	Joint Test Action Group
SED	Soft Error Detection
CCS	Configuration Control System
SPI	Serial Peripheral Interface
AES	Advanced Encryption Standard
SEU	Single Event Upsets
ECC	Error Control Coding
SECDED	Single Error Correcting and Double Error Detecting
UID	Unique ID

Related Documentation

The following documentation is related to this manual:

1. ***DS02001_Logos Family FPGAs Datasheet***
2. ***Fabric_Configuration_User_Guide***
3. ***AN020006_Logos Family FPGAs SEU Application Guide***

Tables of Contents

Tables

Figures

Chapter 1 Overview

Configuration is the process of writing the user's design data (bitstream) into the internal memory of an FPGA. Configuration data can be actively acquired by the chip from an external Flash, or downloaded into the chip via an external processor/controller. Logos Family FPGAs use SRAM cells to store configuration data, which is lost upon power down. Therefore, it is required to reconfigure the FPGA upon each power-up.

Logos Family FPGAs support the following six configuration modes: (For details on the modes supported by different devices, see Chapter 2)

- JTAG mode, compliant with IEEE 1149.1 and IEEE 1532 standards
- Master SPI mode, supporting 1/2/4/8-bit data width
- Master BPI mode, supporting 8/16-bit data width modes
- Slave SPI mode, supporting a data width of 1 bit
- Slave Parallel mode, supporting 8/16/32-bit data width modes
- Slave Serial mode, supporting a data width of 1 bit

In addition, Logos Family FPGAs also provide the following functions:

- Watchdog, supporting time-out detection
- Suppots SEU 1-bit error correction and 2-bit error detection through the internal slave parallel interface
- Configuration bitstream version fallback in the Master BPI or Master SPI mode
- Configuration bitstream compression, which effectively reduces the bitstream size, storage space, and programming time
- Configuration bitstream encryption, which protects customers' design intellectual property against malicious copying
- JTAG mode, which provides a dedicated interface that supports internal debugging and boundary scan testing
- 64-bit UID protection, which writes a UID to each device before leaving the factory

Chapter 2 Description of Configuration Modes

Users can select the configuration mode for Logos Family FPGAs by setting the value of MODE, as shown in Table 2-1. Among them, the JTAG configuration mode has the highest priority, and can be configured by setting MODE to any value. The direction of configuration clock CFG_CLK is determined by the configuration mode. In the master mode, CFG_CLK is an output, provided by the FPGA to an external device storing the bitstream, such as Flash; in the slave mode, CFG_CLK is an input, provided to the FPGA by an external device (such as a microprocessor, CPLD, or another FPGA). This chapter provides a detailed description of the application interfaces for the six configuration modes of Logos Family FPGAs listed in Table 2-1.

Table 2-1 Configuration Modes of Logos Family FPGAs

<table>
<tr><th rowspan="3">S. No.</th><th rowspan="3">Configuration Mode</th><th rowspan="3">bit width</th><th colspan="8">Support Information</th><th rowspan="3">CFG_CLK Direction</th></tr>
<tr><th colspan="2">PGL12G</th><th colspan="2">PGL22G</th><th>PGL22GS</th><th>PGL25G/50G/50H</th><th>PGL100H</th></tr>
<tr><th>LPG144</th><th>FBG256</th><th>FBG256</th><th>MBG324</th><th>LPG176</th><th>ALL</th><th>ALL</th></tr>
<tr><td>1</td><td>JTAG</td><td>1</td><td>√</td><td>√</td><td>√</td><td>√</td><td>√</td><td>√</td><td>√</td><td>Input (TCK)</td></tr>
<tr><td rowspan="4">2</td><td rowspan="4">Master SPI</td><td>1</td><td rowspan="4">√</td><td rowspan="4">√</td><td rowspan="4">√</td><td rowspan="4">√</td><td rowspan="4">√(1)</td><td rowspan="3">√</td><td rowspan="3">√</td><td rowspan="4">Output</td></tr>
<tr><td>2</td></tr>
<tr><td>4</td></tr>
<tr><td>8</td><td>×</td><td>×</td></tr>
<tr><td rowspan="3">3</td><td rowspan="3">Master BPI</td><td>8 (Asynchronous)</td><td rowspan="3">×</td><td rowspan="3">×</td><td rowspan="3">×</td><td rowspan="3">√</td><td rowspan="3">×</td><td rowspan="3">×</td><td rowspan="2">√</td><td rowspan="3">Output</td></tr>
<tr><td>16 (Asynchronous)</td></tr>
<tr><td>16 (Synchronous)</td><td>×</td></tr>
<tr><td>4</td><td>Slave SPI</td><td>1</td><td>√</td><td>√</td><td>√</td><td>√</td><td>×</td><td>×</td><td>×</td><td>Input</td></tr>
<tr><td rowspan="3">5</td><td rowspan="3">Slave Parallel</td><td>8</td><td>√</td><td rowspan="3">√</td><td rowspan="2">√</td><td rowspan="3">√</td><td rowspan="3">×</td><td rowspan="2">√</td><td rowspan="2">√</td><td rowspan="3">Input</td></tr>
<tr><td>16</td><td rowspan="2">×</td></tr>
<tr><td>32</td><td>×</td><td>×</td><td>×</td></tr>
<tr><td>6</td><td>Slave Serial</td><td>1</td><td>√</td><td>√</td><td>√</td><td>√</td><td>×</td><td>√</td><td>√</td><td>Input</td></tr>
</table>

Note: PGL 22 GS Master SPI mode does not support the serial daisy chain.

The functions of configuration pins PGL12G, 22G, and 22GS are defined as follows:

Table 2-2 Function Definitions of Configuration Pins

Item	Property	I/O	Description
CFG_DONE	Dedicated	Bidirectional (open-drain)	Dedicated configuration status pin: (1) As a status output, this pin is driven low before configuration or during the configuration process; (2) After all configuration data is correctly received and the startup timing is initiated, this pin is released; (3) This pin is an open-drain output and should be connected to VCCIOCFG via an external pull-up resistor (recommended 4.7 KΩ).
RST_N	Dedicated	input	Dedicated configuration input pin: (1) Internal weak pull-up, used to restart the configuration process, active-low; (2) When this pin is low, the FPGA enters reset state, and all IOs are in the HighZ state; (3) This pin should be connected to VCCIOCFG via an external pull-up resistor (recommended 4.7 KΩ).
TCK	Dedicated	input	Dedicated JTAG test clock input pin
TMS	Dedicated	input	Dedicated JTAG test mode selection input pin
TDI	Dedicated	input	Dedicated JTAG test data input pin
TDO	Dedicated	output	Dedicated JTAG test data output pin
MODE_2, MODE_1, MODE_0	Multi-function	input	Used for selecting the configuration mode during configuration, the correspondence for MODE[2:0] is as follows: 000: JTAG 001: Master SPI (X1, X2, X4, X8) 010: Master BPI (X8, X16) 011: Slave Serial (X1) 100: Slave Parallel (X8, X16, X32) 101: Slave SPI (X1) 111: Internal Master SPI (X1, X2, X4)
INIT_FLAG_N	Multi-funtion	Bidirectional (open-drain)	Multi-funtion initialization and configuration status pin: (1) When it is low, it indicates that the FPGA's internal CRAM is being cleared, and this pin will be released by internal control upon completion; (2) If this pin is pulled low externally, it will stall the power-on configuration sequence at the end of the initialization process; (3) If this pin is low during configuration, it indicates an internal configuration error occurred; (4) This pin is an open-drain output and should be connected to VCCIOL0 via an external pull-up resistor (recommended 4.7 KΩ).
CFG_CLK	Multi-funtion	input, output	Configuration clock pin: (1) In the slave mode, this pin serves as a clock input to obtain configuration data from external sources; (2) In the master mode, this pin serves as a clock output to obtain configuration data from external sources;

Item	Property	I/O	Description
D[31, 30...1, 0]	Multi-funtion	inpu, output	32-bit configuration data bus input/output pin: (1) In the Master SPI X1 configuration mode, pin D[0], as command output, is connected to the data input of the SPI flash, and pin D[1], as data input, is connected to the data output of the SPI flash. (2) In the Master SPI X2 configuration mode, pin D[1:0] serves as the data bus. (3) In the Master SPI X4 configuration mode, pin D[3:0] serves as the data bus. (4) In the Master SPI X8 configuration mode, pin D[3:0] serves as the data bus for the first SPI FLASH, and pin D[7:4] serves as the data bus for the second SPI FLASH. (5) In the Slave Serial configuration mode, pin D[1] serves as the data bus. (6) In the Slave SPI configuration mode, pin D[0] is for master device output and slave device input, pin D[1] is for the master device input and slave device output, and pin D[3] is for chip hold. (7) In the Master BPI configuration X8 asynchronous mode, pins D[7:0] serve as an 8-bit data bus. (8) In the Master BPI configuration X16 asynchronous/synchronous mode, pins D[15:0] serve as a 16-bit data bus. (9) In the Slave Parallel X8 configuration mode, pins D[7:0] serve as an 8-bit data bus. (10) In the Slave Parallel X16 configuration mode, pins D[15:0] serve as a 16-bit data bus. (11) In the Slave Parallel X32 configuration mode, pins D[31:0] serve as a 32-bit data bus.
CS_N	Multi-funtion	input	Multi-function configuration pin for chip select input. Active low. (1) When it is low, this pin enables the Slave Parallel mode configuration interface. (2) In other configuration modes, this pin is high-impedance. (3) To make this pin continue playing its configuration function after configuration is complete, users need to set the configuration register to preserve its configuration function.
RWSEL	Multi-funtion	input	Multi-function configuration pin for selecting the read/write input in the Slave Parallel configuration mode (high for read and low for write). (1) When it is high, the Slave Parallel configuration mode reads data from the data bus; (2) When it is low, the Slave Parallel configuration mode writes data to the data bus; (3) Read and write can be switched only when CS_N is high. (4) To make this pin continue playing its configuration function after configuration is complete, users need to set the configuration register to preserve its configuration function. (5) In other configuration modes, this pin is high-impedance.
BUSY	Multi-funtion	output	Multi-function configuration pin (1) During readback in the Slave Parallel mode, a high output indicates that the data read from the bus is invalid. (2)To make this pin continue playing its configuration function in the user mode, users need to set the configuration register to preserve its configuration function. (3) In other configuration modes, this pin is high-impedance.
CSO_DOUT	Multi-funtion	output	Multi-function configuration pin needed for cascade. (1) In the Master SPI and X1 modes, this pin serves as cascaded data output;

Item	Property	I/O	Description
			(2) In the Slave Serial configuration mode, this pin serves as cascaded data output; (3) In the Slave Parallel configuration mode, this pin serves as a chip select signal output;
FCS_N	Multi-funtion	output	Multi-function configuration pin, used for the external Master SPI configuration mode. (1) In the Master SPI mode, this pin outputs a chip select signal to external flash when the signal is active-low, and should be connected to VCCIOL0 via an external 4.7K pull-up resistor.
VS1, VS0	Multi-funtion	input	Multi-function configuration pin, used in the Master SPI or internal Master SPI mode. (1) It is used for selecting the bitstream version: 00 for the first set of bitstreams, 01 for the second set, 10 for the third set, 11 for the fourth set; (2) It is internal pull-down resistors during configuration.
IO_STATUS_C	Multi-funtion	input	Multi-function pin, used for inputting signals and controlling the state of all user IOs during the configuration process. (1) "1" keeps all user IOs in a pull-up state during configuration. (2) "0" keeps all user IOs in a tri-state during configuration.
ADR[25:0]	Multi-funtion	output	Multi-function configuration pin, used for outputting addressed in the BPI configuration mode.
BFOE_N	Multi-funtion	output	Multi-function configuration pin, used for providing a low-level output enable control signal for parallel NOR FLASH in the BPI configuration mode. (1) In the BPI configuration mode, this pin should be connected to the flash's output enable input and to VCCIOR0 via a 4.7K resistor.
BADRVO_N	Multi-funtion	output	Multi-function configuration pin, used for providing a low-level address valid control signal for parallel NOR FLASH in the BPI configuration mode. (1) In the BPI configuration mode, if the external FLASH supports address valid signal input, then this pin should be connected to the FLASH's address valid input pin and to VCCIOR0 via a 4.7K resistor. If the external flash does not support address valid signal input, then there is no need to connect this pin.
BFWE_N	Multi-funtion	output	Multi-function configuration pin, used for providing a low-level write enable signal for parallel NOR FLASH in the BPI configuration mode. (1) In the BPI configuration mode, this pin should be connected to the flash's write enable input and to VCCIOR0 via a 4.7K resistor;
BFCE_N	Multi-funtion	output	Multi-function configuration pin, used for providing a low-level chip select control signal for parallel NOR FLASH in the BPI configuration mode; (1) In the BPI configuration mode, should be connected to the flash's chip select input and to VCCIOR0 via a 4.7K resistor.
FCS2_N	Multi-funtion	output	Multi-function configuration pin, used in the external Master SPI X8 configuration mode. (1) In the Master SPI X8 mode, this pin outputs a chip select signal to the external flash when the signal is active-low, and should be connected to VCCIOR0 via an external 4.7K pull-up resistor.
ECCLKIN	Multi-funtion	input	Multi-function configuration pin, used for inputting signals and serves as clock input for the Master configuration mode.

The functions of configuration pins PGL25G, 50G, 50H, and 100H are defined as follows:

Table 2-3 Function Definitions of Configuration Pins

Item	Property	I/O	Description
INIT_FLAG_N	Multi-funtion	Bidirectional (open-drain)	Multi-funtion initialization and configuration status pin: (1) When it is low, it indicates that the FPGA's internal CRAM is being cleared, and this pin will be released by internal control upon completion; (2) If this pin is pulled low externally, it will delay the configuration process; (3) If this pin is low during configuration, it indicates an internal configuration error occurred; (4) This pin is an open-drain output and should be connected to VCCIO2 via an external pull-up resistor (recommended 1-4.7 kΩ).
CFG_DONE	Dedicated	Bidirectional (open-drain)	Dedicated configuration status pin: (1) As a status output, this pin is driven low before configuration or during the configuration process; (2) After all configuration data is correctly received and the startup timing is initiated, this pin is released; (3) This pin is an open-drain output and should be connected to VCCIO2 via an external pull-up resistor (recommended 1–4.7 KΩ).
RST_N	Dedicated	input	Dedicated configuration input pin: (1) Internal weak pull-up, used to restart the configuration process, active-low; (2) When this pin is low, the FPGA enters reset state, and all IOs are in the HighZ state; (3) This pin should be connected to VCCIO2 via an external pull-up resistor (recommended 1–4.7 KΩ).
CFG_CLK	Multi-funtion	inpu, output	Clock configuration pin: (1) In the slave mode, this pin serves as a clock input to obtain configuration data from external sources; (2) In the master mode, this pin serves as a clock output to obtain configuration data from external sources; (3) When the clock is not needed (such as in the JTAG mode), this pin is high-impedance.
TCK	Dedicated	input	Dedicated JTAG test clock input pin, with a built-in weak pull-up resistor.
TMS	Dedicated	input	Dedicated JTAG test mode selection input pin, with a built-in weak pull-up resistor.
TDI	Dedicated	input	Dedicated JTAG test data input pin, with a built-in weak pull-up resistor.
TDO	Dedicated	output	Dedicated JTAG test data output pin, with a built-in weak pull-up resistor.
MODE_1	Multi-funtion	input	Multi-function configuration input pin, used for selecting between master and slave configuration modes: (1) MODE_1=0 (recommended pull-down to ground via a 1K Ohm resistor), master mode; (2) MODE_1=1 (recommended pull-up to VCCIO2 via a 1K Ohm resistor), slave mode.
MODE_0	Multi-funtion	input	Multi-function configuration input pin, used for selecting between parallel and serial configuration modes: (1) MODE_0=0 (recommended pull-down to ground via a 1K Ohm resistor), parallel configuration; (2) MODE_0=1 (recommended pull-up to VCCIO2 via a 1K Ohm resistor), serial configuration.
ECCLK	Multi-funtion	input	Optional external configuration clock input pin in the master mode

Item	Property	I/O	Description
CS_N	Multi-funtion	inpu, output	Multi-function configuration pin: (1) In the Slave Parallel configuration mode, this pin enables the configuration data interface when it is low; (2) In the SPI x1 mode, when this pin is connected to the Slave Data input interface of the SPI Flash, FPGA will send instructions and initial address to the SPI Flash; (3) In the SPI x2 and x4 modes, it is connected to the SPI flash's IO0 as the [0]th bit of the data bus.
CSO_N	Multi-funtion	output	Multi-function configuration pin: (1) In the Slave Parallel configuration mode, this pin serves as cascaded chip select signal output; (2) In the Master SPI mode, this pin serves as chip select signal output and should be connected to VCCIO2 via an external 4.7 KΩ pull-up resistor.
D[0]	Multi-funtion	inpu, output	Multi-function configuration data pins: (1) In the SPI x1 mode, this pin connects to the Slave Data output interface of the SPI Flash, and FPGA receives serial data from the SPI Flash, i.e., Master Input/Slave Output; (2) In the SPI x2 and x4 modes, this pin also serves as the [1]st bit of the SPI data bus, connected to the SPI Flash's IO1; (3) In the Slave Parallel mode, this pin serves as the lowest bit of the data bus; (4) In the Slave Serial mode, this pin serves as data input.
D[1,2]	Multi-funtion	inpu, output	Multi-function configuration data pins: (1) In the SPI x2 and x4 modes, pin D[1] serves as the [2]nd bit connected to SPI flash's IO2, pin D[2] as the [3]rd bit connected to SPI flash's IO3. Both pins should be connected to VCCIO2 via an external 4.7 KΩ pull-up resistor; (2) In the Slave Parallel mode, these pins serve as the [1:2] bits of the data bus.
D[3, 4, 5...15]	Multi-funtion	inpu, output	Multi-function configuration data pins: (1) In the Slave Parallel mode with x8 width, D[7:3] serves as the [7:3]th bit of the data bus; (2) In the Slave Parallel mode with x16 width, D[15:3] serves as the [15:3]th bit of the data bus.
RWSEL	Multi-funtion	input	Multi-function configuration pin, used for selecting the read/write input in the Slave Parallel configuration mode: (1) When it is high, the Slave Parallel configuration mode reads data from the data bus; (2) When it is low, the Slave Parallel configuration mode writes data to the data bus; (3) Read and write can be switched only when CS_N is high.
DOUT_BUSY	Multi-funtion	output	Multi-function configuration pin: (1) During readback in the Slave Parallel mode, this pin indicates the device status. It indicates the data read from the bus is invalid when high; (2) In the Slave Serial configuration mode, this pin serves as cascaded data output, and the data is valid on the falling edge of CFG_CLK; (3) In the SPI configuration mode, this pin serves as cascaded data output, and the data is valid on the falling edge of CFG_CLK.
IO_STATUS_C	Multi-funtion	input	Multi-function input pin, used for controlling whether the pull-up resistors for all user IOs are enabled during the configuration process: (1) When it is set to "0", the internal pull-up resistors for user IOs are enabled before or during configuration;

Item	Property	I/O	Description
			(2) When it is set to "1", the internal pull-up resistors for user IOs are disabled before or during configuration; (3) It is recommended to connect this pin to the corresponding VCCIO0 or ground via a 1 KΩ resistor; (4) This pin must not be left floating before or during configuration.

2.1 JTAG Configuration Mode

The application interface for the JTAG configuration mode is shown in the figure below.

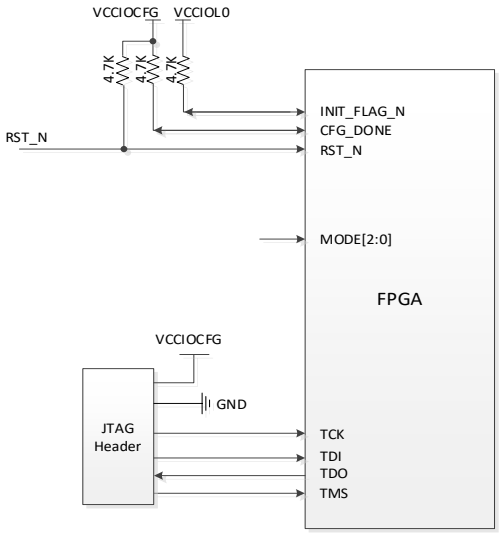

Figure 2-1 Application Interface for the JTAG Configuration Mode of PGL12G, 22G, and 22GS

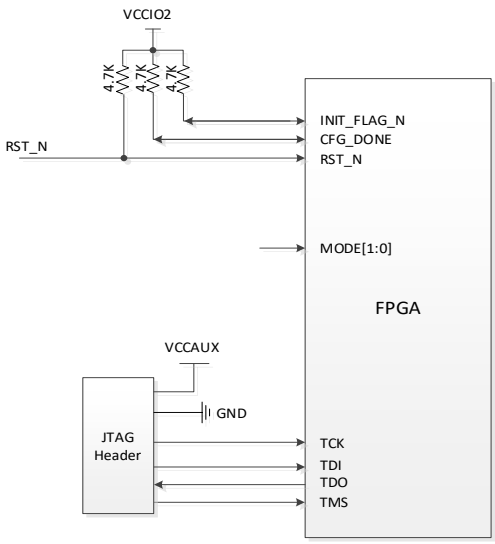

Figure 2-2 Application Interface for the JTAG Configuration Mode of PGL25G, 50G, 50H, and 100H

The configuration logic will be reset when RST_N is low. During initialization, when INIT_FLAG_N is used as an input and connected to a low level, the Logos FPGA will remain in the initialization stage; when used as an output, a high level indicates that the chip initialization ends. If CFG_DONE outputs a high level, it indicates that the chip has entered the user mode, while an external low input will keep the Logos FPGA in the configuration stage.

For PGL12G, 22G, and 22GS, it is recommended to pull up RST_N and CFG_DONE to VCCIOCFG via a 4.7K resistor, and pull up INIT_FLAG_N to VCCIOL0 via a 4.7K resistor. For PGL25G, 50G, 50H, and 100H, it is recommended to pull up RST_N, CFG_DONE, and INIT_FLAG_N to VCCIO2 via a 4.7K resistor.

In the JTAG mode, the TCK should be provided externally; the external can control the transition of the JTAG internal TAP state machine by changing the state of TMS, to select configuration bitstream writing (TDI) or on-die data readback (TDO). In addition to configuration programming, the JTAG interface is also commonly used for internal debugging and boundary scan testing. In the JTAG mode, it is recommended to pull up TDI, TCK, and TMS to VCCIOCFG (PGL12G/22G/22GS) or VCCAUX (PGL25G/50G/50H/100H) via a 15K (±20%) resistor to provide a stable initial input level.

The typical timing for the JTAG programming mode is shown in the figure below.

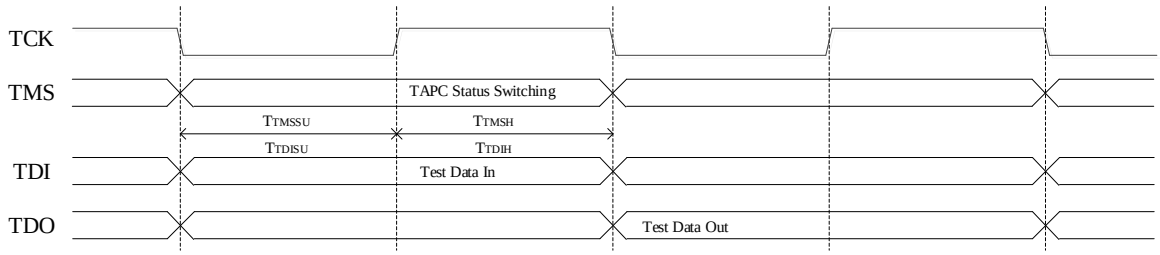

Figure 2-3 Typical Timing for JTAG Programming

Note: For T_{TMSSU}, T_{TDISU}, T_{TMSH}, and T_{TDIH}, see Section 4.7 of the "***DS02001_Logos Family FPGAs Datasheet***".

2.1.1 JTAG Cascade Mode

Multiple devices can be connected using a JTAG daisy chain, as shown in the figure below.

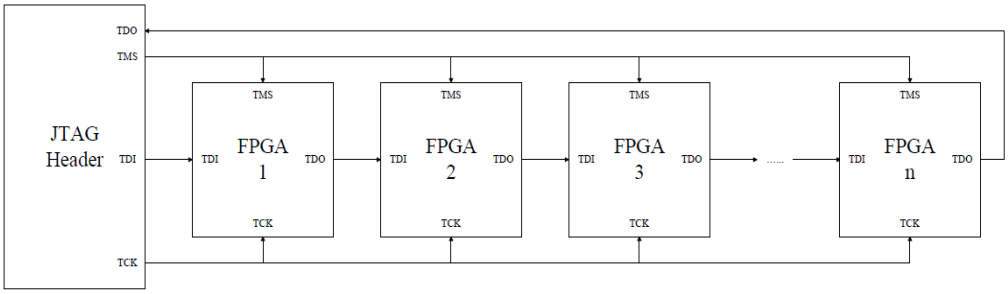

Figure 2-4 JTAG Cascading Application Diagram

In this method, the mode pin should be set to the JTAG mode. After connecting the download cable, the PDS software on the host will scan all the devices on the JTAG chain, and users can choose to program and download the corresponding FPGA.

The TCK and TMS signals connect all the devices on the JTAG chain, so their quality will affect the maximum frequency and reliability of JTAG configuration. Note: During JTAG cascade, only the external Flash of the first-level FPGA can be operated.

2.2 Master SPI Configuration Mode

In the Master SPI mode, bitstreams are usually stored in the external SPI Flash. Power up or apply a low pulse to RST_N to initiate programming. Then the Logos Family FPGAs will actively read the bitstream from the external Flash, and the CCS will automatically start fetching data from address 0 of the Flash in the initial mode (with CFG_CLK as the initial frequency and the rising edge as the data sampling edge, fast read x1 mode). After fetching the bitstream field that sets the SPI mode, the CCS will continue to fetch data using the mode specified in the bitstream (CFG_CLK frequency, data sampling edge, and bit width). Once all the specified bitstreams are fetched from the Flash, FCS_N/FCS2_N (or CSO_N) will be set to 1 to end operation in the Master SPI mode.

2.2.1 PGL12G/22G/22GS

The interface for the Master SPI configuration mode is shown in the figure below:

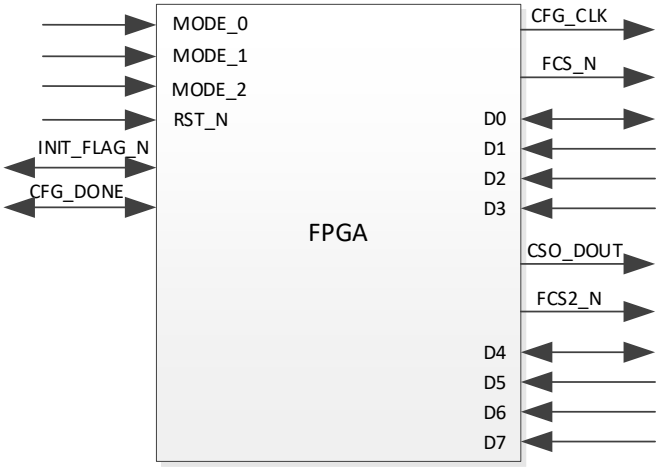

Figure 2-5 Interface for the Master SPI Configuration Mode

The signals of the interface for the Master SPI configuration mode are described in the table below.

Table 2-4 Master SPI Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	O	Configures the clock
MODE[2:0]	Multi-funtion	I	Configuration mode pin 001: Master SPI configuration modes (x1/x2/x4/x8)
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling pins MODE[2:0], INIT_FLAG_N is an input, and configuration can be delayed by keeping it at a low level. After sampling pins MODE[2:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
FCS_N	Multi-funtion	O	SPI Flash chip select signal, active-low
FCS2_N	Multi-funtion	O	Second SPI Flash chip select signal in the x8 mode, active-low
D[7:0]	Multi-funtion	IO	Input-output data bus, sampling data from the rising or falling edge of CFG_CLK D[7:4] is the serial input-output data bus for the second SPI Flash in the x8 mode
CSO_DOUT	Multi-funtion	O	Daisy-chain data output, transmitted on the falling edge of CFG_CLK. PGL22GS does not support daisy chain as it does not have this pin.
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured

Note:

1. The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

To select the Master SPI mode, it is recommended to connect MODE [2:1] to ground via a pull-

down resistor and MODE[0] to the VCCIOL0 power supply via a pull-up resistor.

CFG_CLK is generated and output by the chip internally; monitor pins INIT_FLAG_N and CFG_DONE to judge whether programming is completed; Master SPI supports 1/2/4/8-bit data width modes, which can be set by the software bitstream, and the device selects the bit width based on the bitstream interpretation. For different bit widths, the data bit selection and data input/output direction will be different, as described below:

In the x1 width mode, the FPGA's pin D[0] is connected to the data input of the SPI Flash as a command output, and pin D[1] is connected to the data output of the SPI Flash as a data input. The application is shown in the figure below:

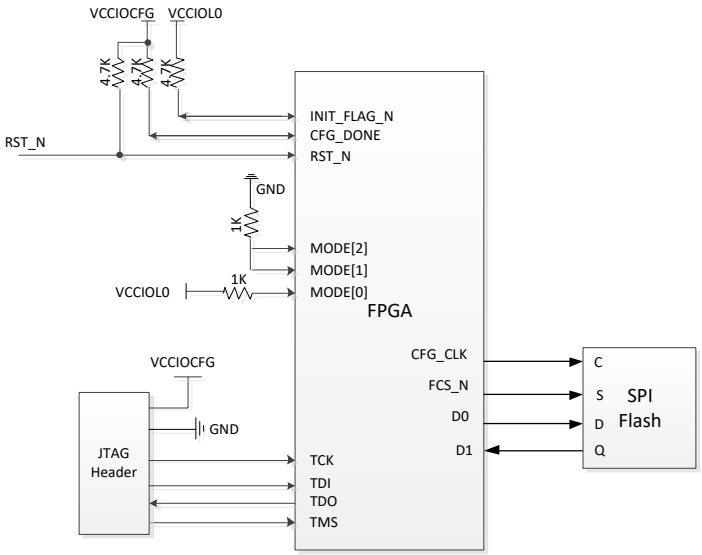

Figure 2-6 Application Diagram of the Master SPI Configuration Mode x1

When the width is x2, pins D[1:0] serve as the data bus. The application is shown in the figure below:

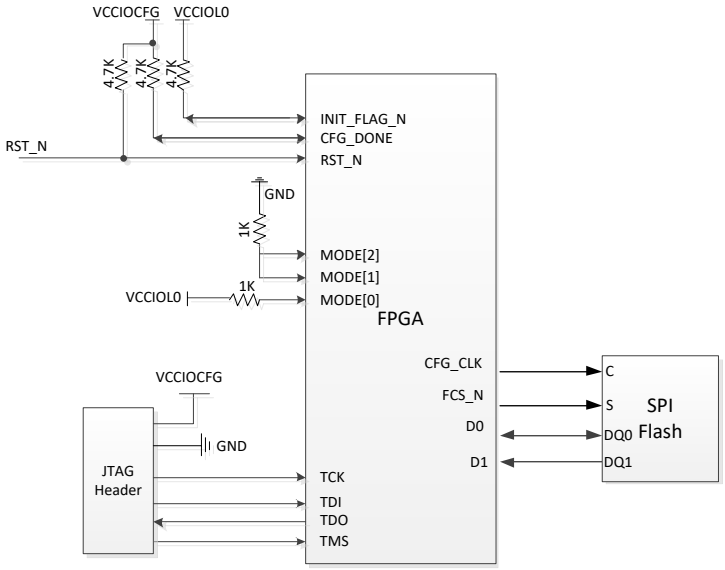

Figure 2-7 Application Diagram of the Master SPI Configuration Mode x2

When the width is x4, pins D[3:0] serve as the data bus. The application is shown in the figure below:

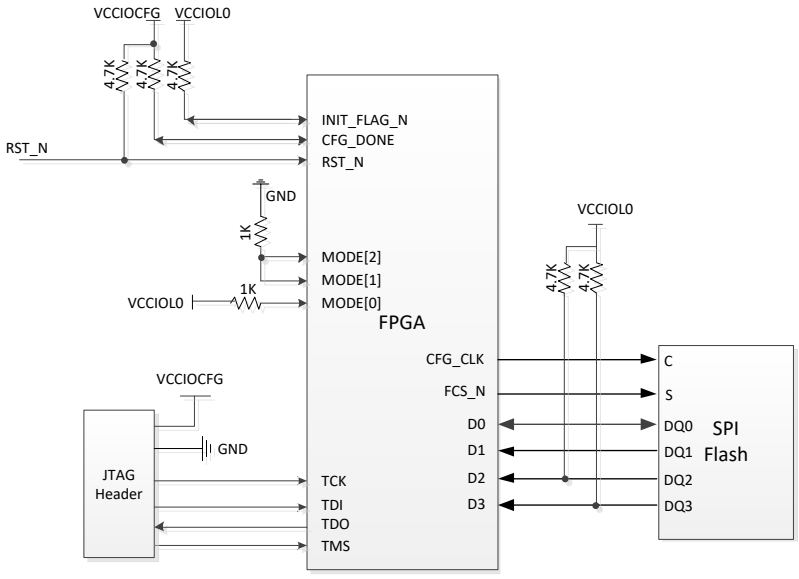

Figure 2-8 Application Diagram of the Master SPI Configuration Mode x4

When the width is x8, pins D[3:0] are the data bus for the first SPI Flash, and pins D[7:4] are the data bus for the second SPI Flash. The application is shown in the figure below:

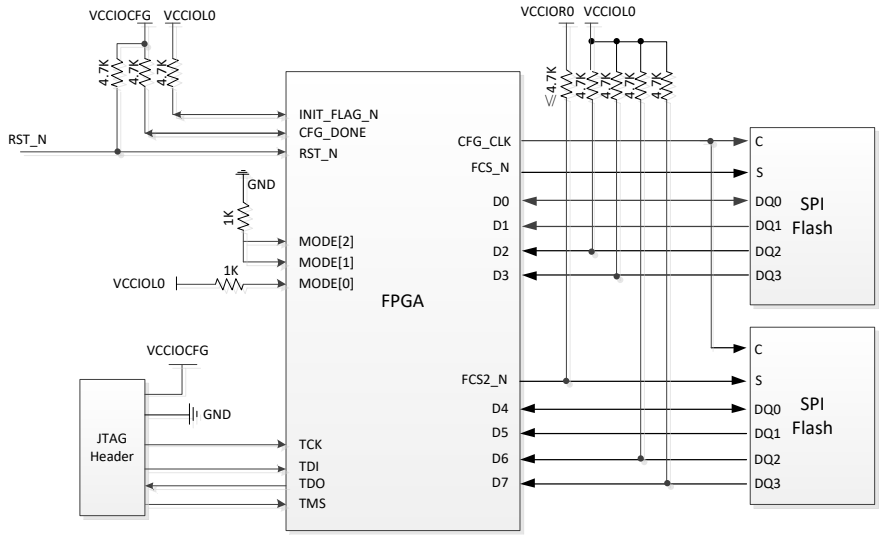

Figure 2-9 Application Diagram of the Master SPI Configuration Mode x8

It is important to keep the VCCIOL0 and VCCIOR0 voltages consistent with the SPI Flash power supply voltage, and the SPI Flash must be powered up before the first CFG_CLK clock rising edge. Only in this way can the FPGA read the Flash data correctly.

The typical timing for the Master SPI programming mode is shown in the figure below (sampling data from the rising edge):

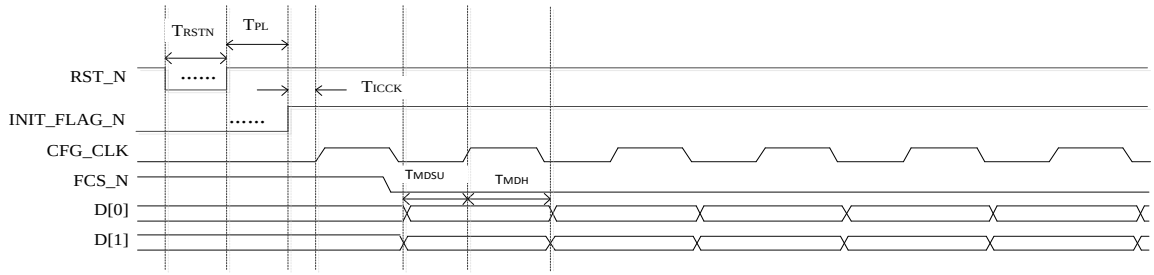

Figure 2-10 Typical Timing for 1bit Master SPI Programming

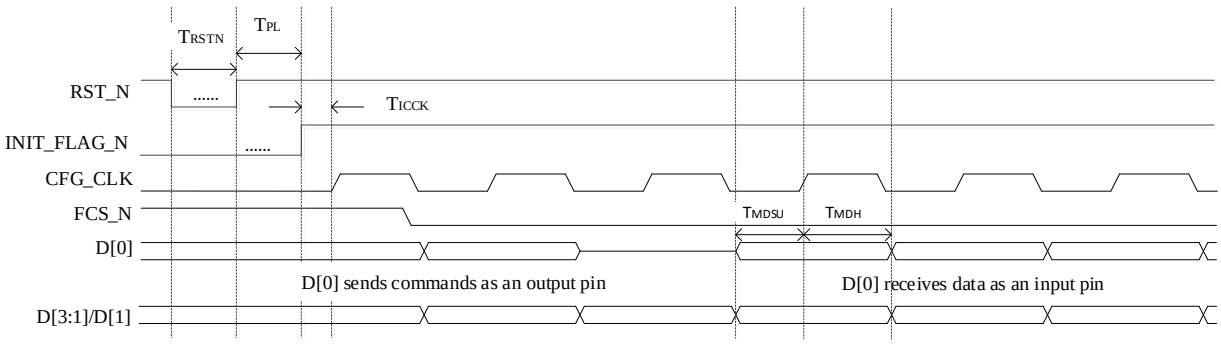

Figure 2-11 Typical Timing Figure for 2/4bit Master SPI Programming

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MDSU}, and T_{MDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.2.1.1 Master SPI Serial Daisy Chain

The Master SPI configuration mode supports SPI serial daisy chain configuration using both Master SPI and Slave Serial modes. In the SPI serial daisy chain, the first device uses the Master SPI mode, while the other devices use the Slave Serial mode. Configure from the last device to the first one in the SPI serial daisy chain. The logic block diagram is shown below:

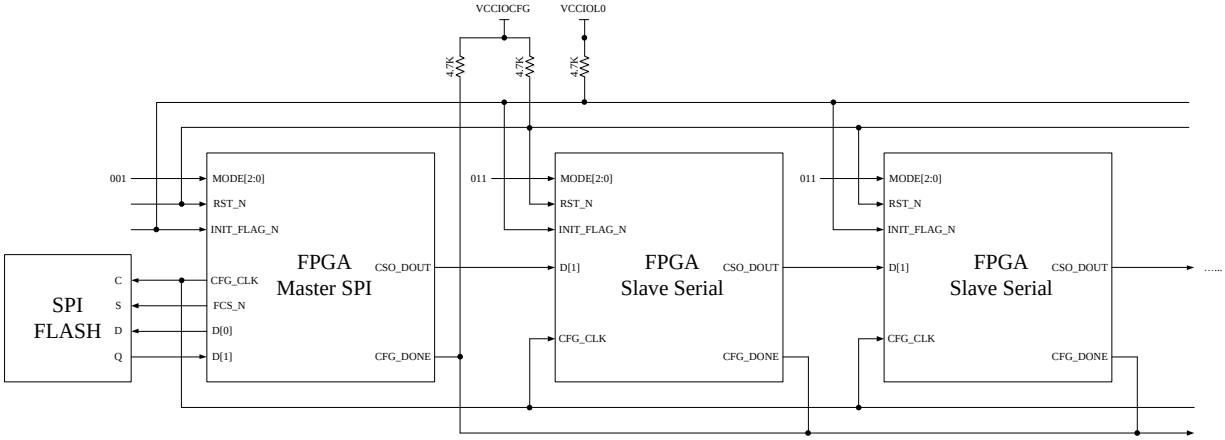

Figure 2-12 Application Diagram of the Master SPI Serial Daisy Chain

2.2.2 PGL25G/50G/50H/100H

The interface for the Master SPI configuration mode is shown in the figure below:

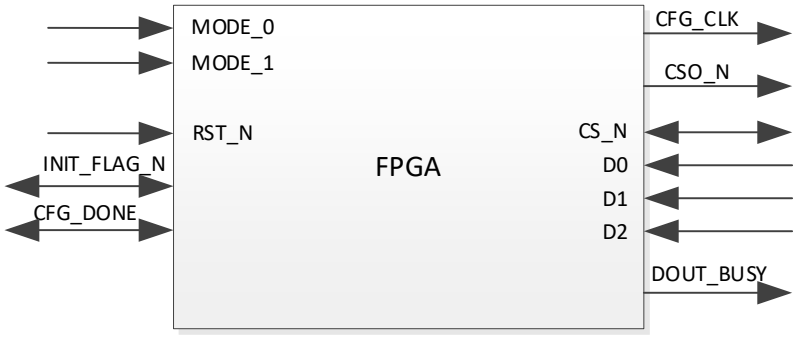

Figure 2-13 Interface for the Master SPI Configuration Mode

The signals of the interface for the Master SPI configuration mode are described in the table below.

Table 2-5 Master SPI Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	O	Configures the clock
MODE[1:0]	Multi-funtion	I	Configuration mode pin 01: Master SPI configuration modes (x1/x2/x4)
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling the MODE[1:0] pins, INIT_FLAG_N is an input, which can delay configuration by maintaining a low level. After sampling pins MODE[1:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
CSO_N	Multi-funtion	O	SPI Flash chip select signal, active-low
CS_N	Multi-funtion	IO	In the x1 mode, it is for master device output and slave device input. In the x2/x4 configuration mode, it is the lowest bit of the data bus, sampling data from the rising or falling edge of CFG_CLK.
D[2:0]	Multi-funtion	I	Serial input-output data bus, sampling data from the rising or falling edge of CFG_CLK In the x1 mode, pin D[0] is for the master device input and slave device output In the x2 mode, pin D[0] is the high input bit for the data bus In the x4 mode, pin D[2:0] are the top 3 high input bits for the data bus
DOUT_BUSY	Multi-funtion	O	Daisy chain data output, transmitted on the falling edge of CFG_CLK
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured

Note:

1. The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

To select the Master SPI mode, it is recommended to connect MODE[1] to ground via a pull-down resistor and MODE[0] to the VCCIO2 power supply via a pull-up resistor.

CFG_CLK is generated and output by the chip internally; monitor pins INIT_FLAG_N and CFG_DONE to judge whether programming is completed; Master SPI supports 1/2/4-bit data width modes, which can be set by the software bitstream, and the device selects the bit width based on the bitstream interpretation. For different bit widths, the data bit selection and data input/output direction will be different, as described below:

In the x1 width mode, the FPGA's pin CS_N is connected to the data input of the SPI Flash as a command output, and pin D[0] is connected to the data output of the SPI Flash as a data input. The application is shown in the figure below:

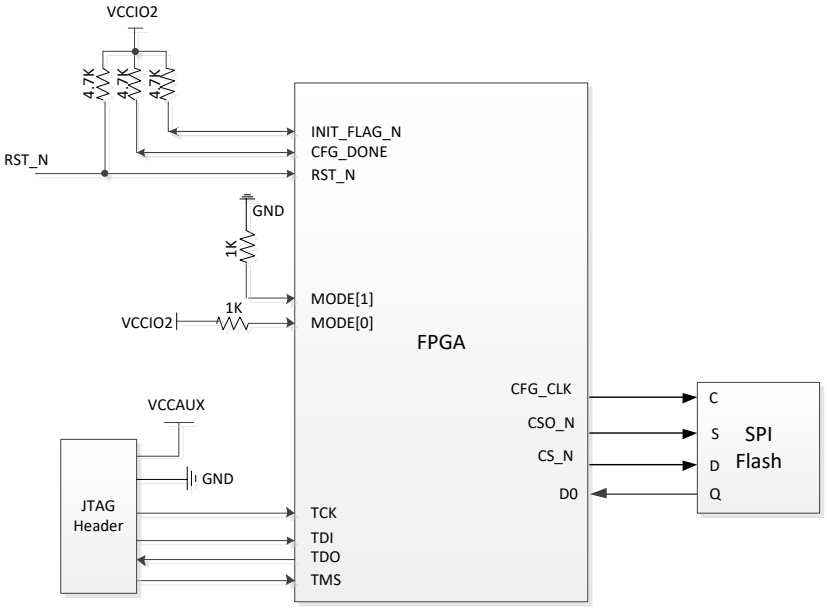

Figure 2-14 Application Diagram of the Master SPI Configuration Mode x1

When the width is x2, pins CS_N and D[0] serve as the data bus. The application is shown in the figure below:

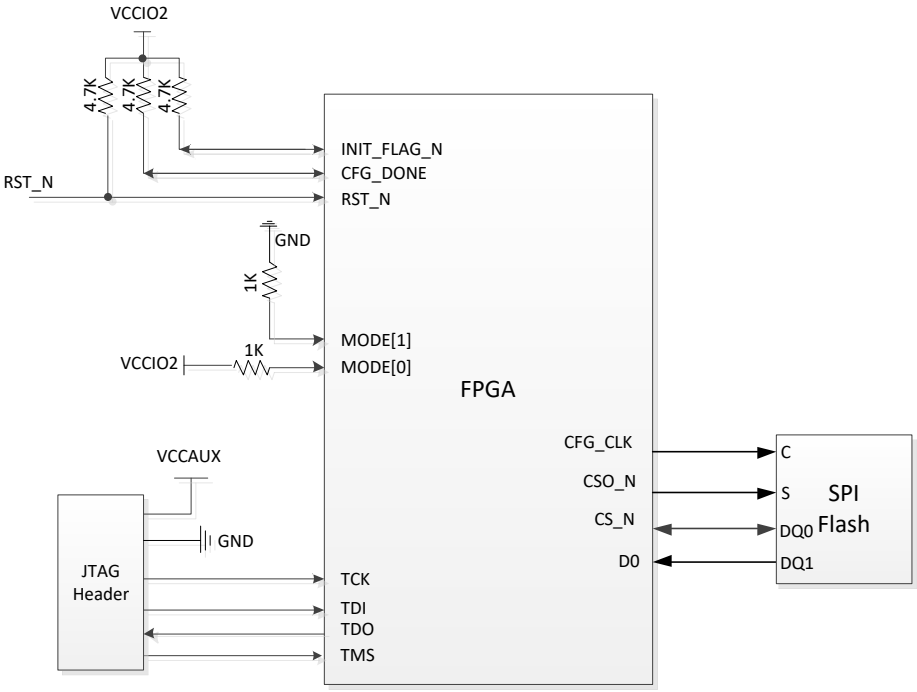

Figure 2-15 Application Diagram of the Master SPI Configuration Mode x2

When the width is x4, pins D[2:0] and CS_N serve as the data bus. The application is shown in the figure below:

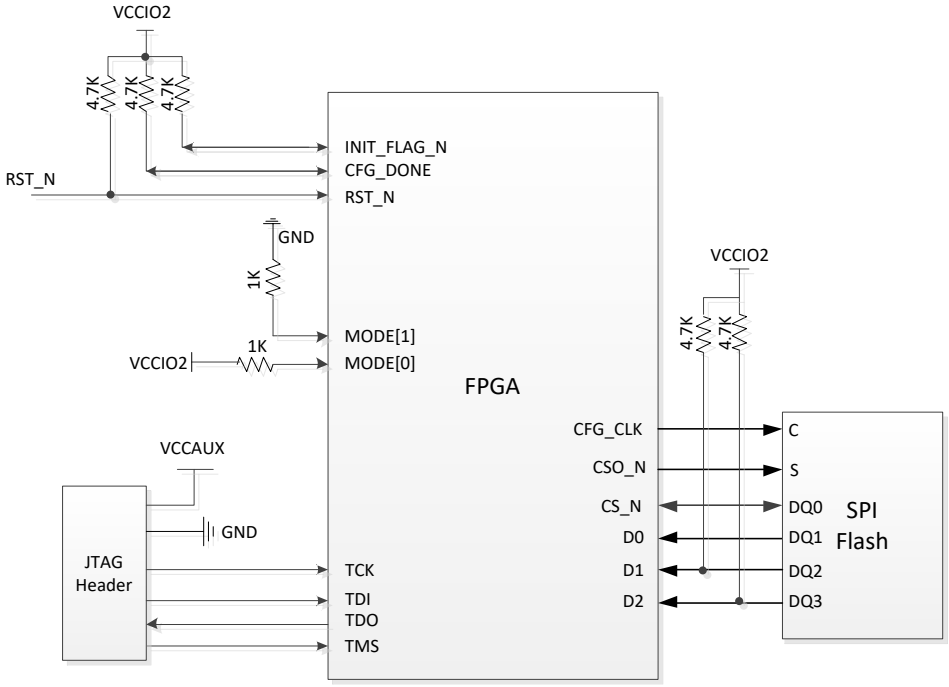

Figure 2-16 Application Diagram of the Master SPI Configuration Mode x4

It is important to keep the VCCIO2 voltage consistent with the SPI Flash power supply voltage, and the SPI Flash must be powered up before the first CFG_CLK clock rising edge. Only in this way can the FPGA read the Flash data correctly.

The typical timing for the Master SPI programming mode is shown in the figure below (sampling data from the rising edge):

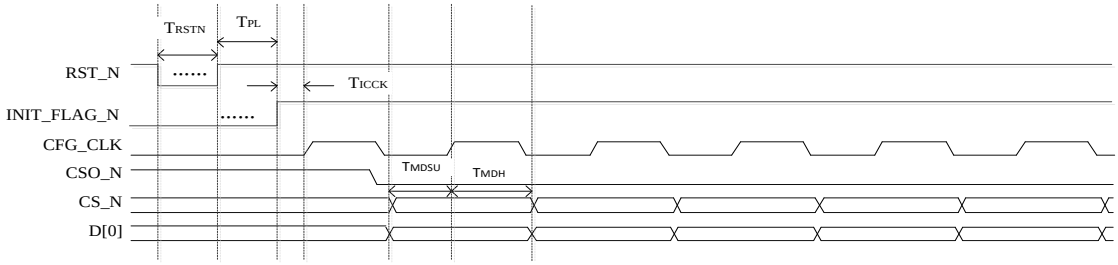

Figure 2-17 Typical Timing for 1bit Master SPI Programming

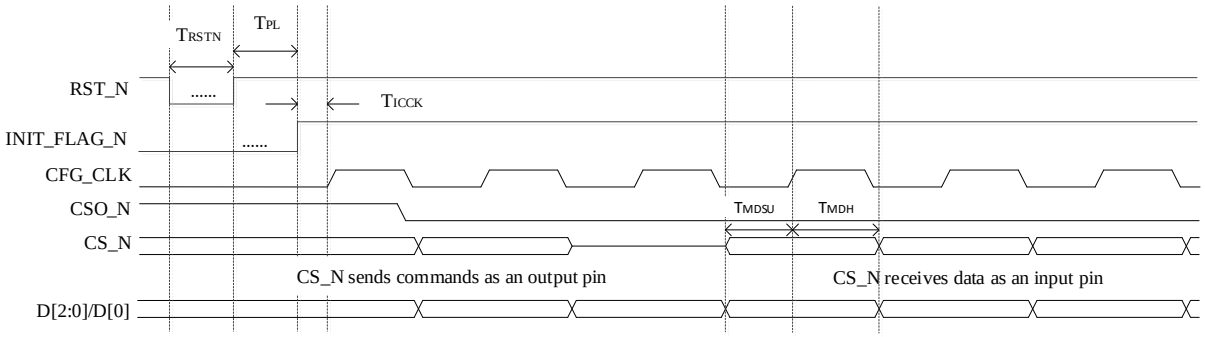

Figure 2-18 Typical Timing for 2/4bit Master SPI Programming

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MDSU}, and T_{MDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.2.2.1 Master SPI Serial Daisy Chain

The Master SPI configuration mode supports SPI serial daisy chain configuration using both Master SPI and Slave Serial modes. In the SPI serial daisy chain, the first device uses the Master SPI mode, while the other devices use the Slave Serial mode. Configure from the last device to the first one in the SPI serial daisy chain. The logic block diagram is shown below:

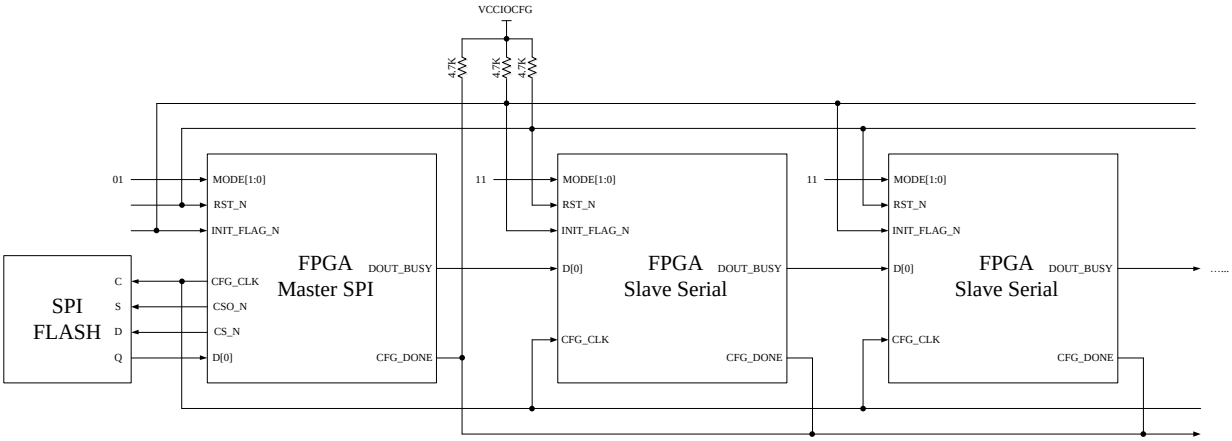

Figure 2-19 Application Diagram of the Master SPI Serial Daisy Chain

2.2.3 List of Supported SPI Flash Models

The following table shows some SPI Flash models supported by the Logos Family FPGAs.

Table 2-6 Supported SPI Flash Models

Model	**Manufacturer**	**Density**
AT45DB081E	ADESTO	8Mb
S25FL64	CYPRESS	64Mb
S25FL128	CYPRESS	128Mb
S25FL256	CYPRESS	256Mb

Model	Manufacturer	Density
S25FL512	CYPRESS	512Mb
EN25Q80C	ESMT	8Mb
FH25VQ80D	FENTECH	8Mb
M25P16	Micron	16Mb
M25P32	Micron	32Mb
M25P64	Micron	64Mb
M25P128	Micron	128Mb
N25Q32	Micron	32Mb
N25Q64	Micron	64Mb
N25Q128	Micron	128Mb
N25Q256	Micron	256Mb
N25Q512	Micron	512Mb
MD25Q80C	GigaDevice	8Mb
GD25Q80C	GigaDevice	8Mb
GD25Q32C	GigaDevice	32Mb
GD25Q64C	GigaDevice	64Mb
GD25Q128C	GigaDevice	128Mb
GD25Q256D	GigaDevice	256Mb
GD25Q512C	GigaDevice	512Mb
XM25QH16B	XMC	16Mb
XM25QH32B	XMC	32Mb
XM25QH64A	XMC	64Mb
XM25QH128A	XMC	128Mb
XM25QH256B	XMC	256Mb
W25Q40CL	WINBOND	4Mb
W25Q80	WINBOND	8Mb
W25Q16	WINBOND	16Mb
W25Q32	WINBOND	32Mb
W25Q64Q	WINBOND	64Mb
W25Q64M	WINBOND	64Mb
W25Q128Q	WINBOND	128Mb
W25Q128M	WINBOND	128Mb
W25Q256	WINBOND	256Mb
W25Q512	WINBOND	512Mb
IS25LP016D	ISSI	16Mb
IS25LP032D	ISSI	32Mb
IS25LP064D	ISSI	64Mb
IS25LP128D	ISSI	128Mb
IS25LP256D	ISSI	256Mb

Model	Manufacturer	Density
IS25LP512D	ISSI	512Mb
IS25WP016D	ISSI	16Mb
IS25WP032D	ISSI	32Mb
IS25WP064D	ISSI	64Mb
IS25WP128D	ISSI	128Mb
IS25WP256D	ISSI	256Mb
IS25WP512D	ISSI	512Mb
MX25L32	MXIC	32Mb
MX25L64	MXIC	64Mb
MX25L128	MXIC	128Mb
MX25L256	MXIC	256Mb
MX25L512	MXIC	512Mb
ZB25VQ16	ZBIT	16Mb
ZB25VQ32	ZBIT	32Mb
SM25QH256M	SSMEC	256Mb

Note: The models listed in the table have been tested. Other models should undergo testing and verification before use. When selecting a Flash, consider the actual storage space requirements (such as the actual bitstream size; see Table 3-3 for the ordinary bitstream size of Logos devices).

2.3 Master BPI Configuration Mode

2.3.1 PGL22G

The Master BPI configuration interface is shown in the figure below:

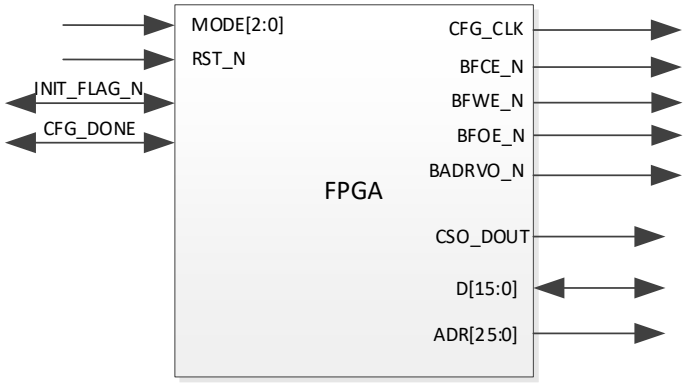

Figure 2-20 Interface for the Master BPI Configuration Mode

The signals of the interface for the Master BPI configuration mode are described in the table below.

Table 2-7 Master BPI Configuration Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low

Item	Property	I/O	Description
CFG_CLK	Multi-funtion	O	Configures the clock
MODE[2:0]	Multi-funtion	I	Configuration mode pin 010: Master BPI configuration modes (asynchronous x8/asynchronous x16/synchronous x16)
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling pins MODE[2:0], INIT_FLAG_N is an input, and configuration can be delayed by keeping it at a low level. After sampling pins MODE[2:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
BFCE_N	Multi-funtion	O	BPI Flash chip select, active-low
BFWE_N	Multi-funtion	O	BPI Flash write enable, active-low
BFOE_N	Multi-funtion	O	BPI Flash output enable, active-low
BADRVO_N	Multi-funtion	O	BPI Flash address valid, active-low; if the external flash does not support address valid signal input, then connection is not required.
D[15:0]	Multi-funtion	IO	Parallel input/output data bus
ADR[25:0]	Multi-funtion	O	BPI Flash 26-bit address bus, byte addressing
CSO_DOUT	Multi-funtion	O	Daisy chain chip select output. The daisy chain chip select output of the previous device connects to the Slave Parallel mode chip select input of the next device.
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured

Note:

1. The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

To select the Master BPI mode, it is recommended to connect MODE[2] and MODE[0] to ground via pull-down resistors, and connect MODE[1] to the VCCIOL0 power supply via a pull-up resistor. The application is shown in the figure below:

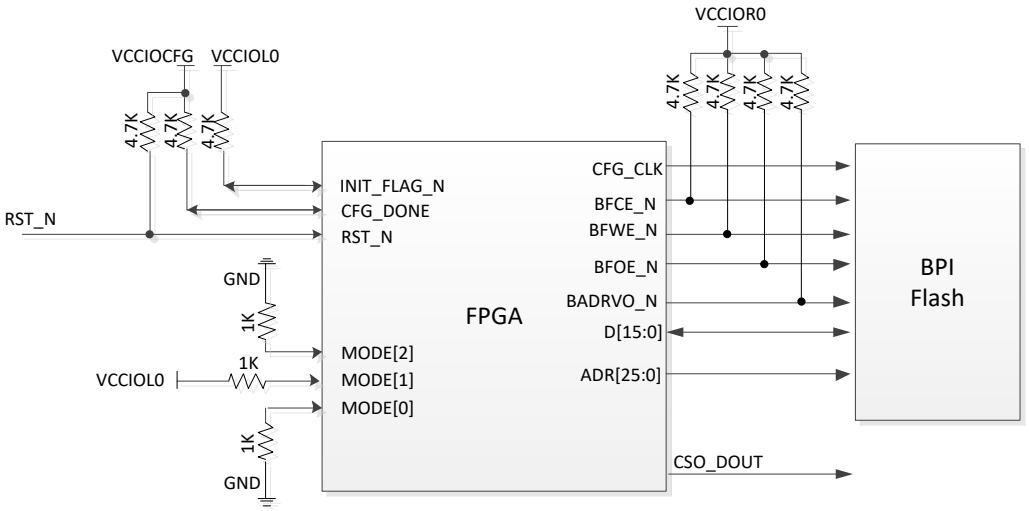

Figure 2-21 Application Diagram of the Master BPI Configuration Mode

The Master BPI mode supports asynchronous read, asynchronous page read, and synchronous read.

2.3.1.1 Asynchronous Read

In the asynchronous mode, the FPGA does not need to provide a clock to the Flash, but it still generates and samples signals via CFG_CLK. The FPGA generates control signals such as Flash chip select on the falling edge of CFG_CLK to send operation codes, data, and addresses. Depending on the settings, the FPGA samples data from the rising or falling edge of CFG_CLK. When the clock frequency is not higher than 25 MHz, data is sampled from the rising edge of CFG_CLK; when the clock frequency is higher than 25 MHz, data is sampled from the falling edge of CFG_CLK. The asynchronous read timing is shown below:

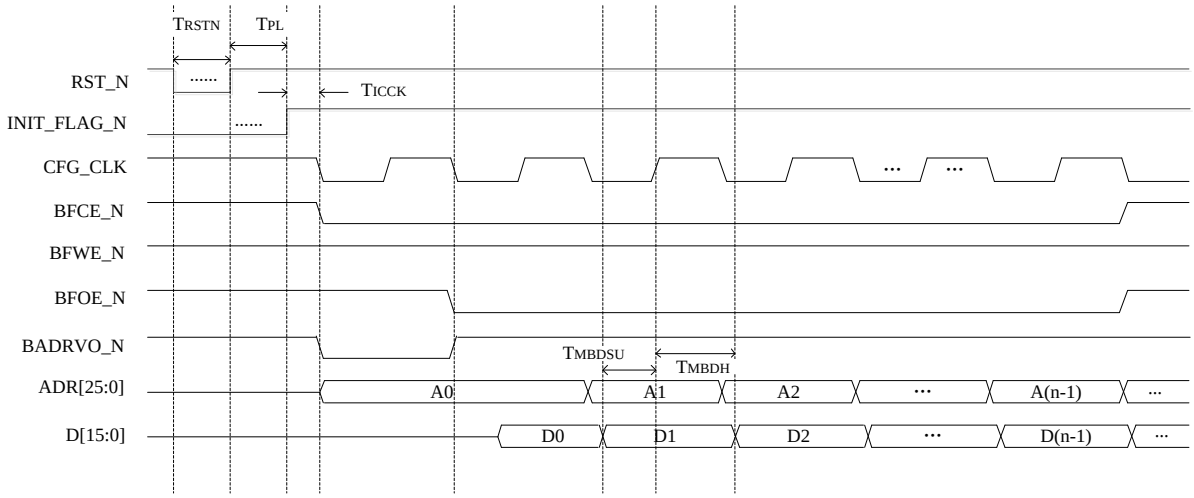

Figure 2-22 Master BPI Asynchronous Read Timing

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MBDSU}, and T_{MBDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

After the Master BPI mode is selected, the CCS will automatically fetch data starting from the 0 address of the BPI Flash by using the asynchronous read mode. Once the SBPI register is fetched, if it is set to the asynchronous read mode, the CCS will fetch bitstream specified by register IRSTCTRLR or pins VS[1:0] from the BPI Flash in the asynchronous read mode. After all the specified bitstreams are fetched from Flash, BFCE_N will be set to 1 to end operation in the Master BPI mode.

During remote upgrade, multifunction configuration, and version fallback, if the SBPI register is set to the asynchronous read mode, the CCS will fetch the bitstream specified by register IRSTCTRLR or pins VS[1:0] from the BPI Flash in the asynchronous read mode. After all the specified bitstreams are fetched from Flash, BFCE_N will be set to 1 to end operation in the Master BPI mode.

2.3.1.2 Asynchronous Page Read

Take a page with 4 bytes/half-words as an example. It takes longer to read the first byte/half-word of each page than the rest ones within the page. Set the number of clocks for reading the first byte to 1–4 through software. The timing is shown in the figure below:

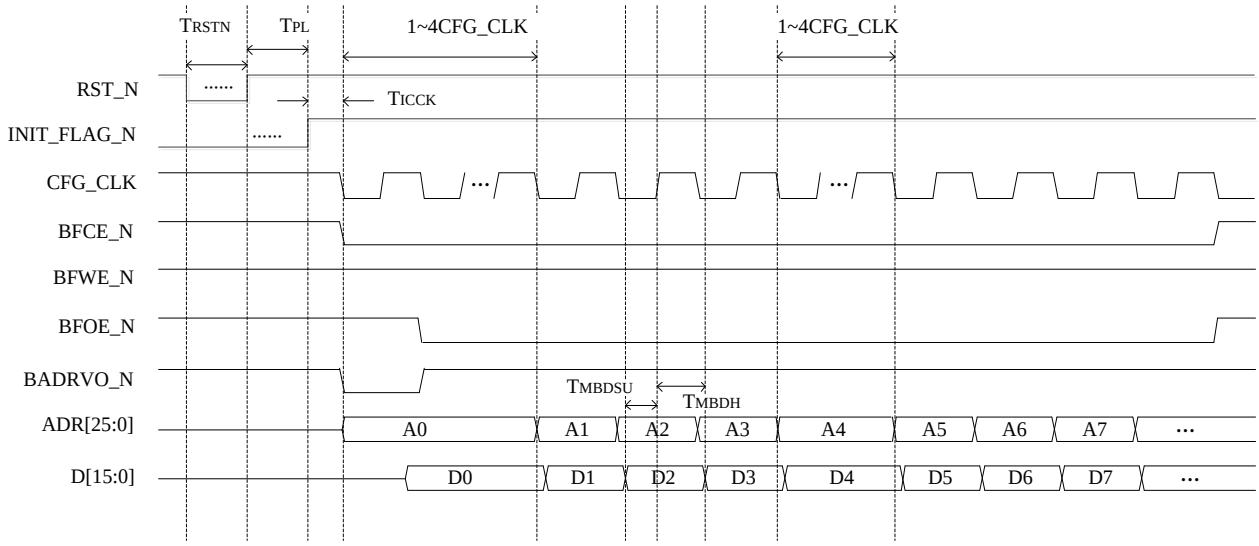

Figure 2-23 Master BPI Asynchronous Page Read Timing

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MBDSU}, and T_{MBDH}, see Section 4.7 of the "DS02001 Logos Family FPGAs Datasheet".

2.3.1.3 Synchronous Read

The FPGA generates control signals such as Flash chip select on the falling edge of CFG_CLK to send operation codes, data, and addresses.

Flash samples the operation codes and addresses from the rising edge of CFG_CLK.

During synchronous readback, Flash generates data on the rising edge of CFG_CLK. Depending on the settings, the FPGA samples data from the rising or falling edge of CFG_CLK. When the clock frequency is not higher than 25M, data is sampled from the falling edge of CFG_CLK; when the clock frequency is higher than 25M, data is sampled from the rising edge of CFG_CLK. The synchronous read timing is shown below:

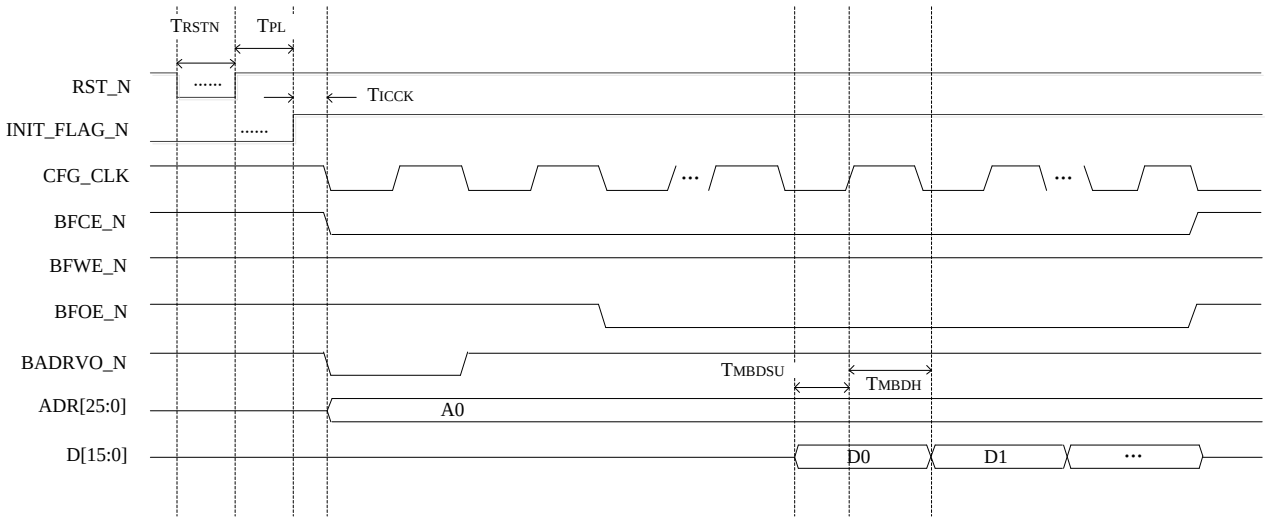

Figure 2-24 Master BPI Synchronous Read Timing

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MBDSU}, and T_{MBDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

After the Master BPI mode is selected, the CCS will automatically fetch data starting from the 0 address of the BPI Flash by using the asynchronous read mode. Once the SBPI register is fetched, if it is set to synchronous read mode, the CCS will first send a program read configuration register instruction to BPI Flash, and then fetch the bitstream specified by register IRSTCTRLR or pins VS[1:0] in the synchronous read mode. After all the specified bitstreams are fetched from Flash, BFCE_N will be set to 1 to end operation in the Master BPI mode.

During remote upgrade, multi-boot, and version fallback, if the SBPI register is set to synchronous read mode, the CCS will first send a program read configuration register instruction to the BPI Flash, and then fetch the bitstream specified by register IRSTCTRLR or pins VS[1:0] in the synchronous read mode. After all the specified bitstreams are fetched from Flash, BFCE_N will be set to 1 to end operation in the Master BPI mode.

2.3.2 PGL100H

The Master BPI configuration interface is shown in the figure below:

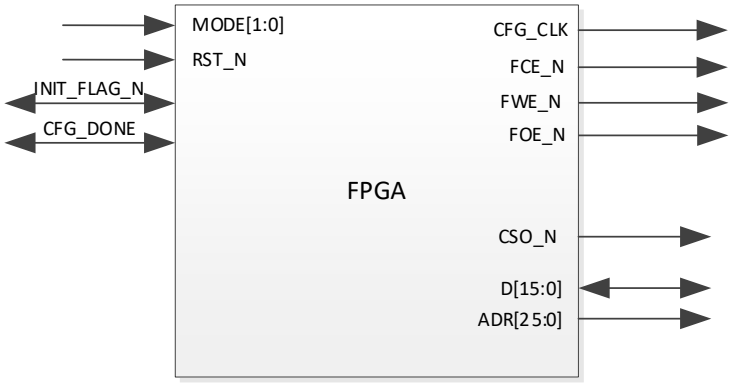

Figure 2-25 Interface for the Master BPI Configuration Mode

The signals of the interface for the Master BPI configuration mode are described in the table below.

Table 2-8 Master BPI Configuration Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	O	Configures the clock
MODE[1:0]	Multi-funtion	I	Configuration mode pin 00: Master BPI configuration modes (asynchronous x8/asynchronous x16)
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling pins MODE[1:0], INIT_FLAG_N is an input, and configuration can be delayed by keeping it at a low level. After sampling pins MODE[1:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
BFCE_N	Multi-funtion	O	BPI Flash chip select, active-low
BFWE_N	Multi-funtion	O	BPI Flash write enable, active-low
BFOE_N	Multi-funtion	O	BPI Flash output enable, active-low
D[15:0]	Multi-funtion	IO	Parallel input/output data bus
ADR[25:0]	Multi-funtion	O	BPI Flash 26-bit address bus, byte addressing
CSO_N	Multi-funtion	O	Daisy chain chip select output. The daisy chain chip select output of the previous device connects to the Slave Parallel mode chip select input of the next device.

Item	Property	I/O	Description
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured

Note:

1. The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

To select the Master BPI mode, it is recommended to connect MODE[1] and MODE[0] to ground via pull-down resistors. The application is shown in the figure below:

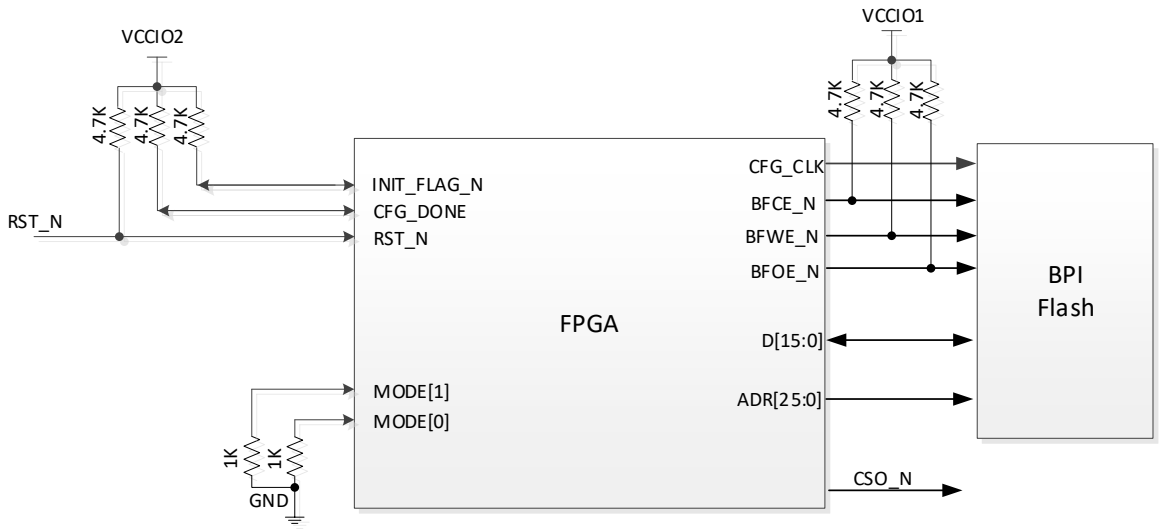

Figure 2-26 Application Diagram of the Master BPI Configuration Mode

The Master BPI mode supports asynchronous read, asynchronous page read, and synchronous read.

2.3.2.1 Asynchronous Read

In the asynchronous mode, the FPGA does not need to provide a clock to the Flash, but it still generates and samples signals via CFG_CLK. The FPGA generates control signals such as Flash chip select on the falling edge of CFG_CLK to send operation codes, data, and addresses. Depending on the settings, the FPGA samples data from the rising or falling edge of CFG_CLK. When the clock frequency is not higher than 25 MHz, data is sampled from the rising edge of CFG_CLK; when the clock frequency is higher than 25 MHz, data is sampled from the falling edge of CFG_CLK. The asynchronous read timing is shown below:

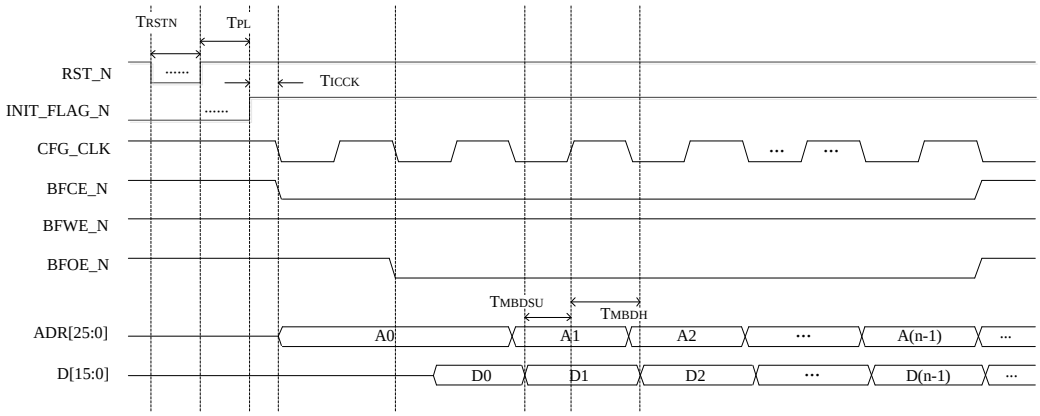

Figure 2-27 Master BPI Asynchronous Read Timing

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MBDSU}, and T_{MBDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

After the Master BPI mode is selected, the CCS will automatically fetch data starting from the 0 address of the BPI Flash by using the asynchronous read mode. Once the SBPI register is fetched, if it is set to the asynchronous read mode, the CCS will fetch bitstream specified by register IRSTADRR from the BPI Flash in the asynchronous read mode. After all the specified bitstreams are fetched from Flash, BFCE_N will be set to 1 to end operation in the Master BPI mode.

During remote upgrade and version fallback, if the SBPI register is set to the asynchronous read mode, the CCS will fetech the bitstream specified by register IRSTADRR from the BPI Flash in the asynchronous read mode. After all the specified bitstreams are fetched from Flash, BFCE_N will be set to 1 to end operation in the Master BPI mode.

2.3.2.2 Asynchronous Page Read

Take a page with 4 bytes/half-words as an example. It takes longer to read the first byte/half-word of each page than the rest ones within the page. Set the number of clocks for reading the first byte to 1–4 through software. The timing is shown in the figure below:

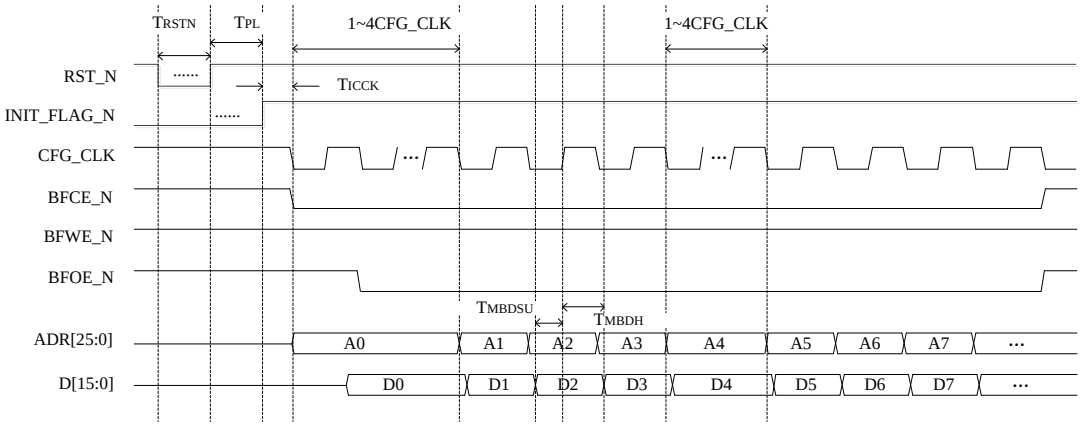

Figure 2-28 Master BPI Asynchronous Page Read Timing

Note: For T_{RSTN}, T_{PL}, T_{ICCK}, T_{MBDSU}, and T_{MBDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.4 Slave SPI Configuration Mode

Only PGL12G/22G supports the Slave SPI configuration mode. The interface for the Slave SPI configuration mode is shown in the figure below:

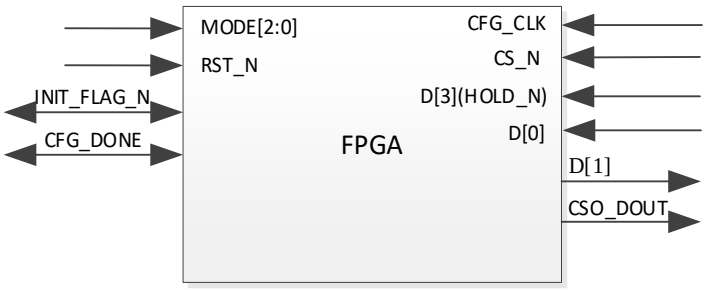

Figure 2-29 Slave SPI Configuration Interface

The signals of the interface for the Slave SPI configuration mode are described in the table below.

Table 2-9 Slave SPI Configuration Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	I	Configures the clock
MODE[2:0]	Multi-funtion	I	Configuration mode pin 101: Slave SPI configuration mode (x1)
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling pins MODE[2:0], INIT_FLAG_N is an input, and configuration can be delayed by keeping it at a low level. After sampling pins MODE[2:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)

Item	Property	I/O	Description
CS_N	Multi-funtion	I	FPGA chip select signal, active-low, The FPGA samples the CS_N on rising CFG_CLK edges, with setup and hold times of 3ns and 1ns respectively.
D[3] (HOLD_N)	Multi-funtion	I	Hold signal, active-low, used for pausing communication between the master device and the FPGA
D[1]	Multi-funtion	O	Serial output data bus
D[0]	Multi-funtion	I	Serial input data bus
CSO_DOUT	Multi-funtion	O	Daisy chain data output, transmitted on the falling edge of CFG_CLK
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured

Note:

(1) The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

In the Slave SPI configuration mode, pin D[0] is for master device output and slave device input, pin D[1] is for the master device input and slave device output, and pin D[3] is for chip hold.

To select the Slave SPI configuration mode, it is recommended to pull up MODE[2] and MODE[0] to VCCIOL0 via resistors, and connect MODE[1] to ground via a pull-down resistor. The application of the Slave SPI configuration mode is shown in the figure below:

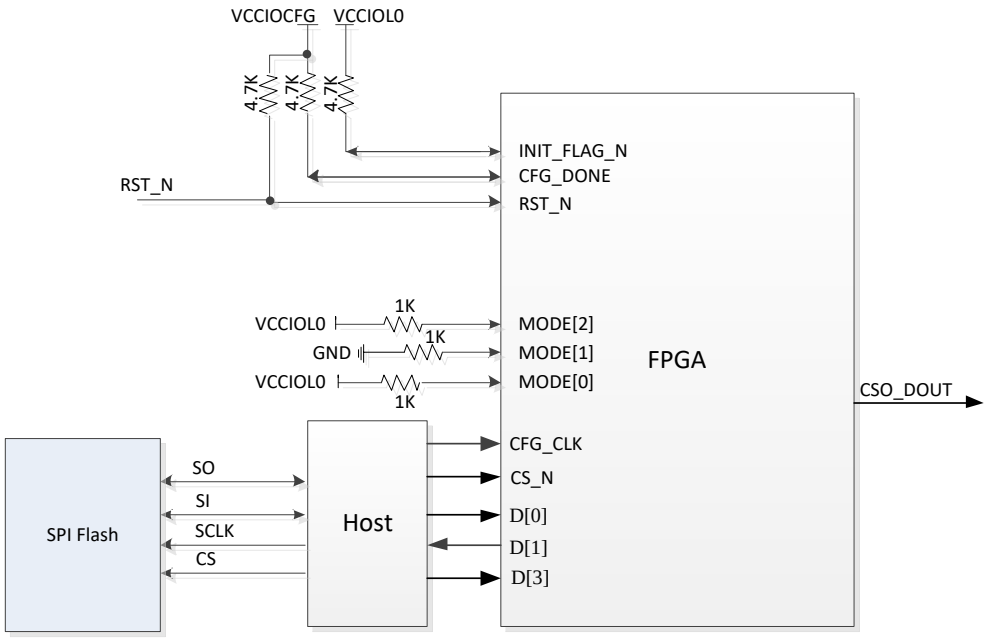

Figure 2-30 Application Diagram of the Slave SPI Mode

In the Slave SPI mode, multiple chips on the board can be powered up and started via a master control chip (Host), which can be a microprocessor, CPLD, or another FPGA; in the Slave SPI

mode, CFG_CLK is provided externally.

In this mode, programming can be initiated by powering up or applying a low pulse to RST_N;

Monitor pins INIT_FLAG_N and CFG_DONE to judge whether programming is completed;

The typical timing for the Slave SPI programming mode is shown in the figure below.

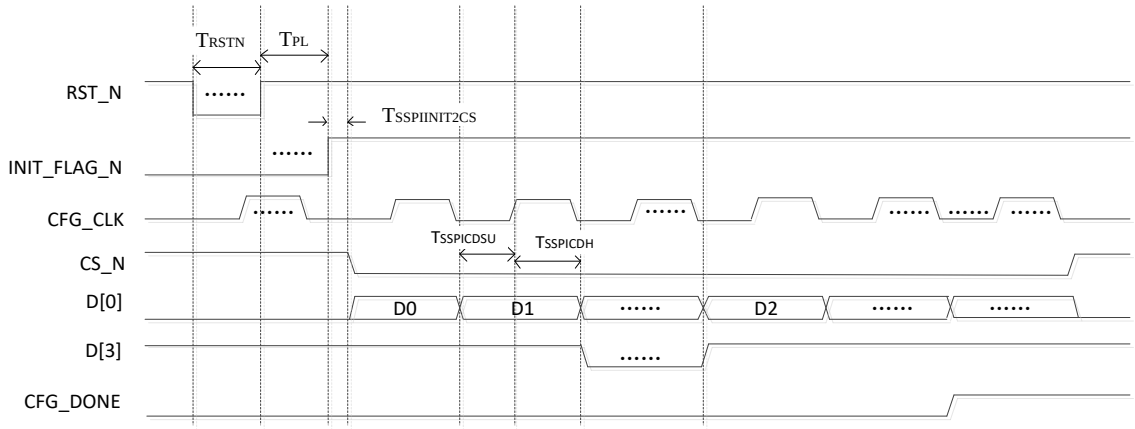

Figure 2-31 Slave SPI Configuration Timing

Note: For T_{RSTN}, T_{PL}, $T_{SSPIINIT2CS}$, $T_{SSPICDSU}$, and $T_{SSPICDH}$, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.4.1 Instruction Set

Table 2-10 Slave SPI Instruction Set Related to Configuration

Instruction	**Description**	**Op Code**
NOP	No Op	FF
RDID	Read IDCODE	A1
RDSR	Read the status register	A3
PROGRAM	Configure the bitstream	50
WREN	Write enable	51
WRDIS	Write disabled	52
ERASE	Reset the FPGA	60

2.4.2 Configuration/Reconfiguration Process

4. Input value 3'b101 to configuration pins MODE[2:0], and keep the value unchanged throughout the configuration process.
5. Power-up
6. When the value of INIT_FLAG_N becomes 1, write the RDID instruction to read the device IDCODE
7. Check the value of IDCODE

8. If IDCODE does not match, terminate the operation. If IDCODE matches, write the WREN instruction
9. Write the PROGRAM instruction, and load the bitstream.
10. After the bitstream transmission is completed, if the Slave SPI interface is released for user in the user mode (Select Persist Pin is not set to Persist Slave SPI Pin in the software), check the values of INIT_FLAG_N and CFG_DONE to conclude the operation. If the Slave SPI interface is persisted as a configuration interface in the user mode, write the WRDIS instruction.
11. Write the RDSR instruction to read the device status register

The FPGA can be cold-booted (by setting RST_N to 0) at any time after power-up. After a cold boot, users can proceed to step 3 to reconfigure the FPGA.

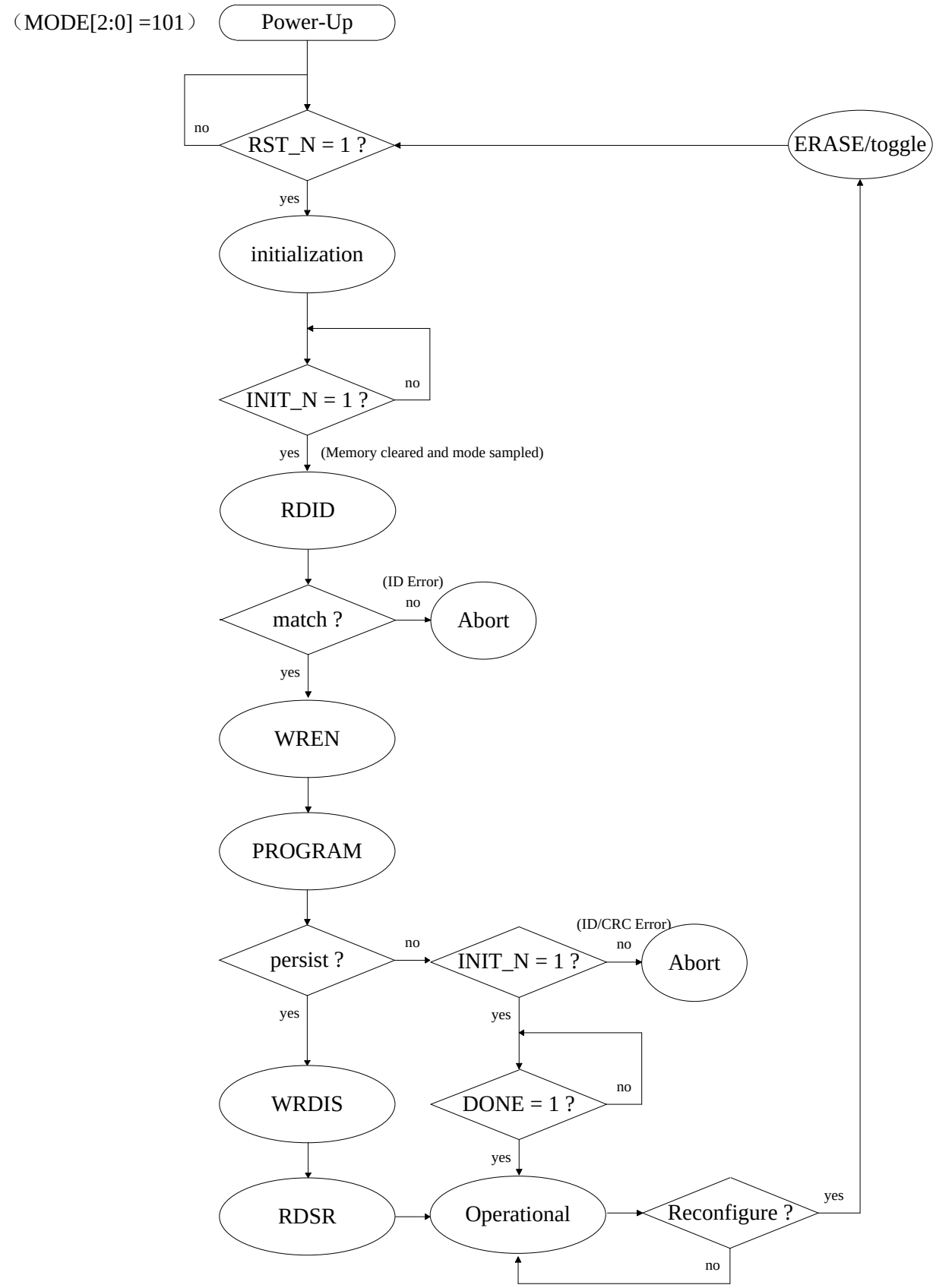

Figure 2-32 Configuration/Reconfiguration Process

- The CS_N signal needs to be pulled high for at least one cycle between instructions, all starting with the highest bit. The timing for the programming process is shown in the figure below:

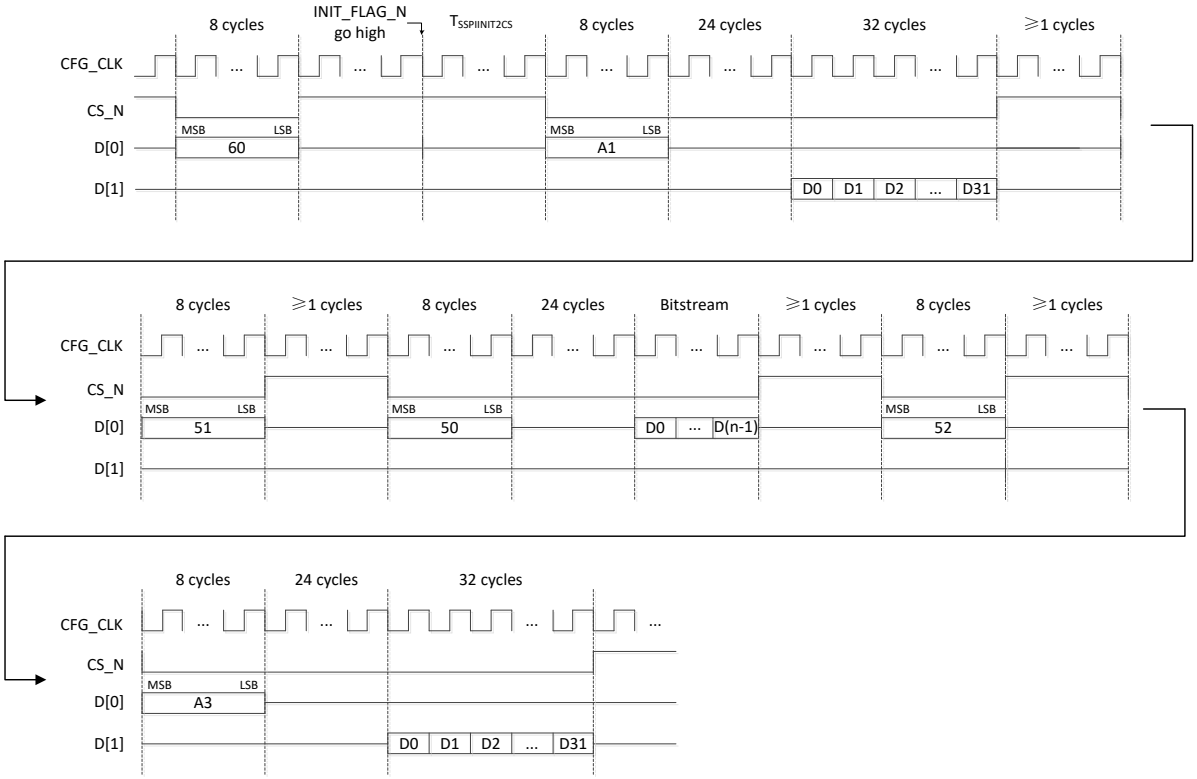

Figure 2-33 Programming Process Timing

Note: For $T_{SSPIINIT2CS}$, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.5 Slave Serial Configuration Mode

2.5.1 PGL12G/22G

The interface for the Slave Serial configuration mode is shown in the figure below:

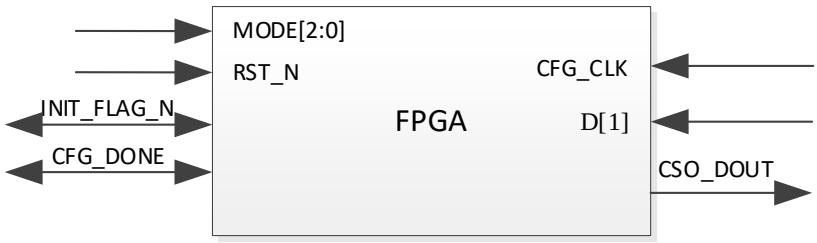

Figure 2-34 Slave Serial Configuration Interface

The signals of the interface for the Slave Serial configuration mode are described in the table below.

Table 2-11 Slave Serial Configuration Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	I	Configures the clock

Item	Property	I/O	Description
MODE[2:0]	Multi-funtion	I	Configuration mode pin 011: Slave Serial mode
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling pins MODE[2:0], INIT_FLAG_N is an input, and configuration can be delayed by keeping it at a low level. After sampling pins MODE[2:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured
D[1]	Multi-funtion	I	Serial input data bus, sampling data from the rising edge of CFG_CLK for self-configuration, and sampling data from the falling edge during cascade for the next-level configuration.
CSO_DOUT	Multi-funtion	O	Daisy chain data output, transmitted on the falling edge of CFG_CLK

Note:

1 The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

To select the Slave Serial mode, it is recommended to connect MODE[2] to ground via a pull-down resistor, and connect MODE[1:0] to VCCIOL0 via a pull-up resistor. The application of the Slave Serial configuration mode is shown in the figure below:

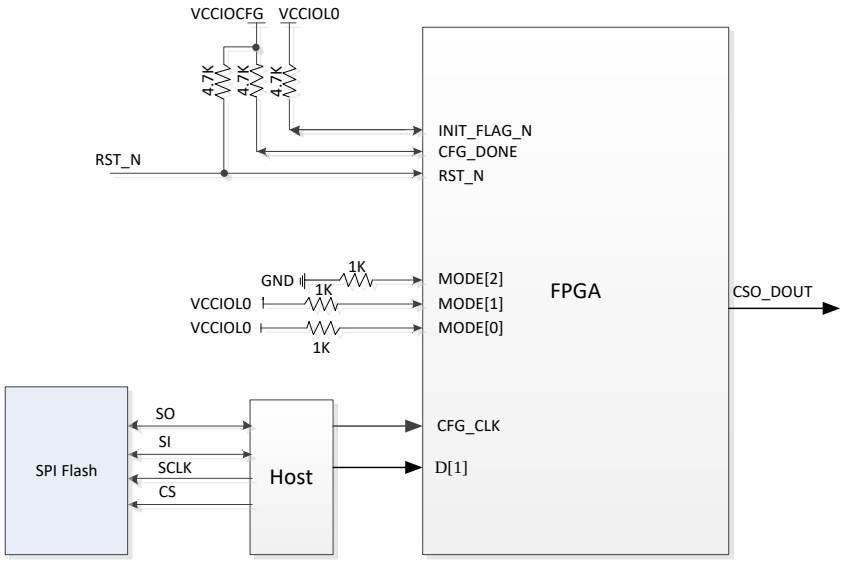

Figure 2-35 Application Diagram of the Slave Serial Mode

In the Slave Serial configuration mode, the power-up, start, and bitstream loading of multiple chips on the board can be controlled via a master control chip (Host). The Host can be a microprocessor, CPLD, or another FPGA. In this mode, programming can be initiated by powering up or applying a

low pulse to RST_N. Monitor pins INIT_FLAG_N and CFG_DONE to judge whether programming is completed.

When the Host sends a bitstream to the device, if wait for PLL Lock is not enabled (disabled by default in software), the device will release control of CFG_DONE during a clock cycle between the 100 NOP type 1 packet headers at the end of the bitstream, and CFG_DONE will go high due to the external pull-up resistor. After CFG_DONE is pulled high, the remaining filler bitstream is used to provide a clock for waking up the device, so it is necessary to ensure that the bitstream is completely tranmitted before terminating the clock. If wait is enabled, the clock cannot be terminated before CFG_DONE is pulled high, and at least 100 clocks must be provided continuously after CFG_DONE is pulled high.

When using serial daisy-chain configuration, each device will sample D[1] from the rising edge of CFG_CLK for its own configuration and also sample D[1] from the falling edge to output configuration for the next device. Therefore, during cascaded configuration, the Host needs to consider both the rising and falling edges to meet the setup and hold time requirements.

The typical timing for the Slave Serial programming mode is shown in the figure below.

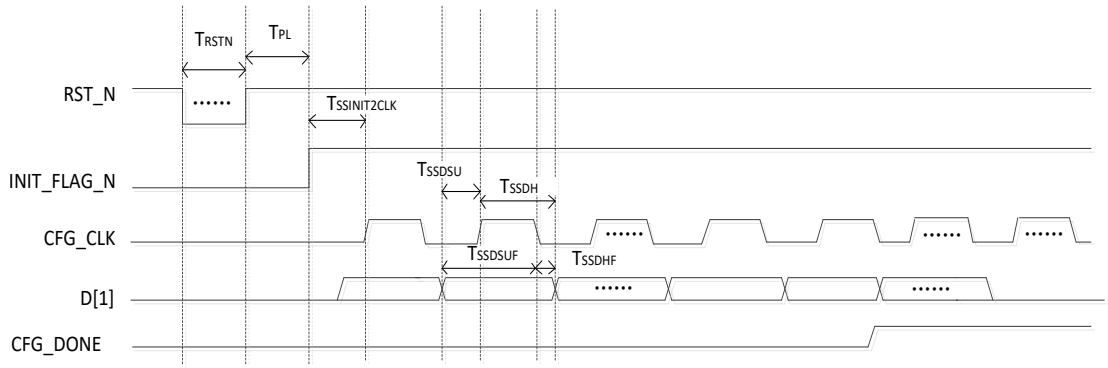

Figure 2-36 Typical Timing for Slave Serial Programming

Note: For T_{RSTN}, T_{PL}, $T_{SSINIT2CLK}$, T_{SSDSU}, T_{SSDH}, T_{SSDSUF}, and T_{SSDHF}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.5.1.1 Slave Serial Daisy Chain

Serial daisy chain is a typical application of the Slave Serial configuration mode. All devices can be configured in a serial cascade using the Slave Serial mode. Configure from the last (furthest) device to the first one in the chain during cascade configuration. Combine several bitstreams using the Fabric Configuration tool into a cascaded bitstream.

Through serial cascade, multiple cascaded chips on the board can be configured via a master control

chip (Host). Connect them as shown in the figure below.

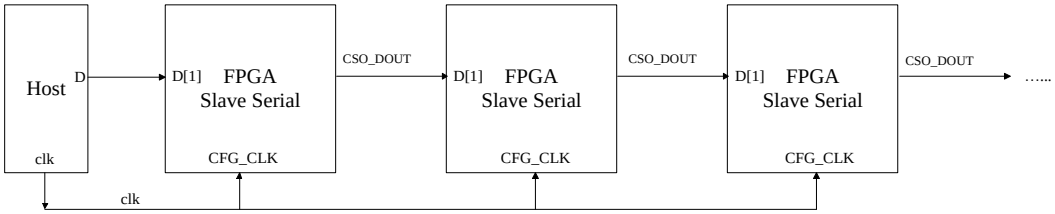

Figure 2-37 Serial Daisy Chain Connection in the Slave Serial Configuration Mode

When adopting the serial daisy chain, be sure to:

Meet the primary stage timing

As shown in Figure 2-36, the timing sent by the Host must meet the setup and hold time requirements for sampling D[1] from the rising and falling edges of CFG_CLK.

Meet the subsequent stage timing

The primary stage timing can be met by adjusting the Host output. The subsequent stage inputs all come from the previous stage, with adjustments mainly involving delays introduced by PCB traces, etc. As shown in the figure below, it is necessary to adjust delay parameters $T_{CLKDELAY}$ and $T_{DOUTDELAY}$ to meet the setup and hold time requirements of the subsequent stage devices for both edges.

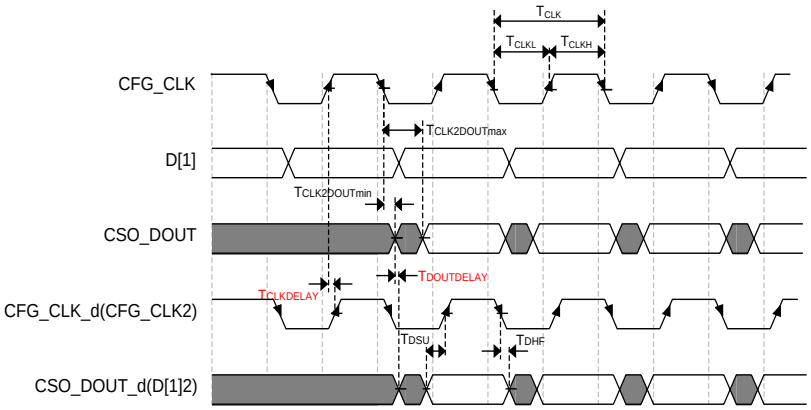

Figure 2-38 Timing Adjustment Example–Subsequent Stage

Note: For $T_{SSCLK2DOUT}$, T_{SSDSU}, and T_{SSDHF}, see Section 4.7 of the "***DS02001_Logos Family FPGAs Datasheet***".

The above figure shows that if the setup time for the rising edge is met when sending data on the falling edge, then the setup time for the falling edge will certainly be met; if the hold time for the falling edge is met, then the hold time for the rising edge will also be met. Therefore, $T_{CLKDELAY}$ and $T_{DOUTDELAY}$ must satisfy the following formula (unit: ns):

$$\begin{cases} T_{\mathrm{DSU}} = (T_{\mathrm{CLK}} + T_{\mathrm{CLKDELAY}}) - (T_{\mathrm{CLKH}} + T_{\mathrm{CLK2DOUTmax}} + T_{\mathrm{DOUTDELAY}}) \geq 2 \\ T_{\mathrm{DHF}} = T_{\mathrm{CLK2DOUTmin}} + T_{\mathrm{DOUTDELAY}} - T_{\mathrm{CLKDELAY}} \geq 1 \end{cases}$$

i.e.,

$$9.5 - T_{\mathrm{CLKL}} \leq T_{\mathrm{CLKDELAY}} - T_{\mathrm{DOUTDELAY}} \leq 1$$

2.5.2 PGL25G/50G/50H/100H

The interface for the Slave Serial configuration mode is shown in the figure below:

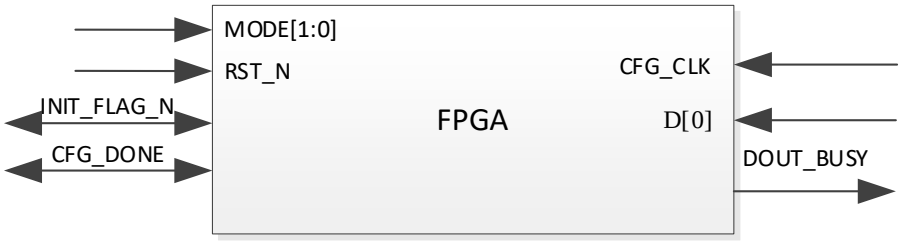

Figure 2-39 Slave Serial Configuration Interface

The signals of the interface for the Slave Serial configuration mode are described in the table below.

Table 2-12 Slave Serial Configuration Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	I	Configures the clock
MODE[1:0]	Multi-funtion	I	Configuration mode pin 11: Slave Serial mode
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling the MODE[1:0] pins, INIT_FLAG_N is an input, which can delay configuration by maintaining a low level. After sampling pins MODE[1:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users (1).
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured
D[0]	Multi-funtion	I	Serial input data bus, sampling data from the rising edge of CFG_CLK for self-configuration, and sampling data from the falling edge during cascade for the next-level configuration.
DOUT_BUSY	Multi-funtion	O	Daisy chain data output, transmitted on the falling edge of CFG_CLK

Note:

1. The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

To select the Slave Serial mode, it is recommended to connect pins MODE[1:0] to VCCIO2 via a pull-up resistor. The application of the Slave Serial configuration mode is shown in the figure below:

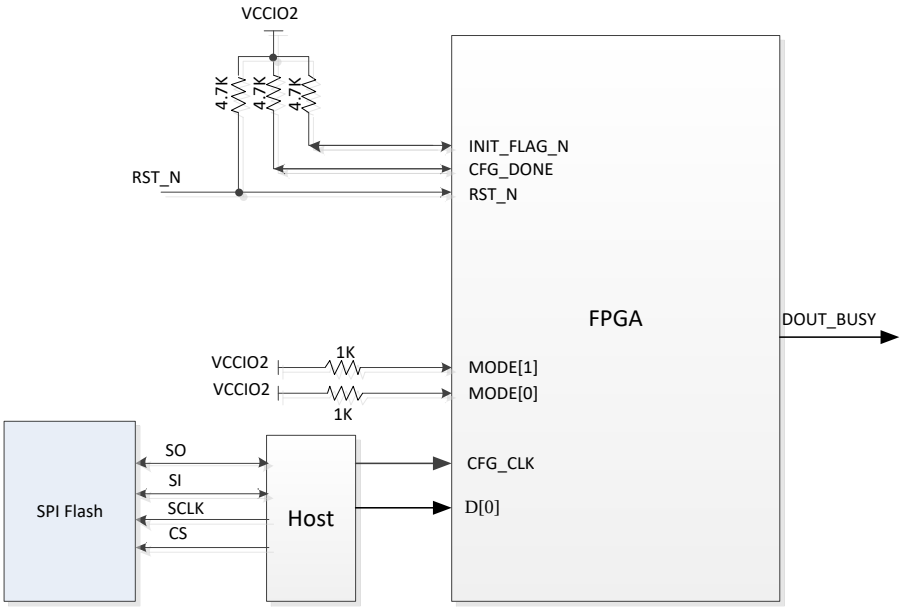

Figure 2-40 Application Diagram of the Slave Serial Mode

In the Slave Serial configuration mode, the power-up, start, and data loading of multiple chips on the board can be controlled via a master control chip (Host). The Host can be a microprocessor, CPLD, or another FPGA. In this mode, programming can be initiated by powering up or applying a low pulse to RST_N. Monitor pins INIT_FLAG_N and CFG_DONE to judge whether programming is completed.

When the Host sends a bitstream to the device, if wait for PLL Lock is not enabled (disabled by default in software), the device will release control of CFG_DONE during a clock cycle between the 100 NOP type 1 packet headers at the end of the bitstream, pulling CFG_DONE high via the external pull-up resistor. After CFG_DONE is pulled high, the remaining filler bitstream is used to provide a clock for waking up the device, so it is necessary to ensure that the bitstream is completely tranmitted before terminating the clock. If wait is enabled, the clock cannot be terminated before CFG_DONE is pulled high, and at least 100 clocks must be provided continuously after CFG_DONE is pulled high.

When using serial daisy-chain configuration, each level of device will sample D[0] from the rising edge of CFG_CLK for its own configuration and also sample D[0] from the falling edge to output configuration for the next level. Therefore, during cascaded configuration, the Host needs to consider both the rising and falling edges to meet the setup and hold time requirements. The typical timing for the Slave Serial programming mode is shown in the figure below.

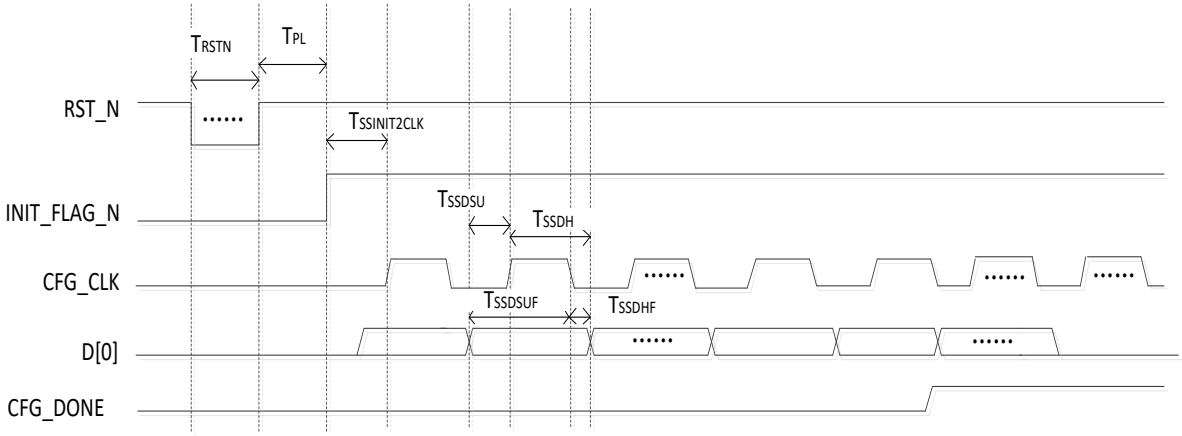

Figure 2-41 Typical Timing for Slave Serial Programming

Note: For T_{RSTN}, T_{PL}, $T_{SSINIT2CLK}$, T_{SSDSU}, T_{SSDH}, T_{SSDSUF}, and T_{SSDHF}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.5.2.1 Slave Serial Daisy Chain

Serial daisy chain is a typical application of the Slave Serial configuration mode. All devices can be configured in a serial cascade using the Slave Serial mode. Configure from the last (furthest) device to the first one in the chain during cascade configuration. Combine several bitstreams using the Fabric Configuration tool into a cascaded bitstream.

Through serial cascade, multiple cascaded chips on the board can be configured via a master control chip (Host). Connect them as shown in the figure below.

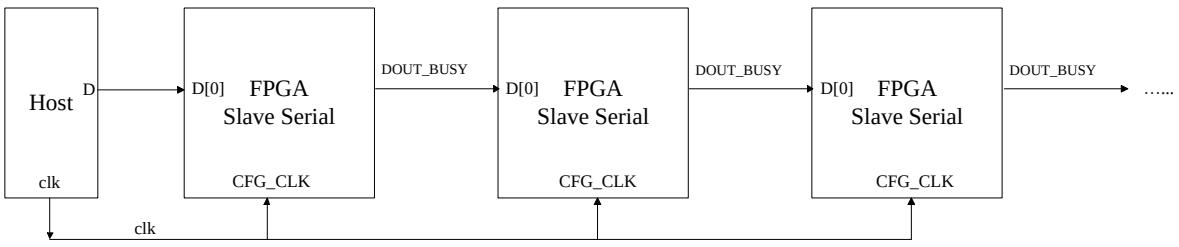

Figure 2-42 Serial Daisy Chain Connection in the Slave Serial Configuration Mode

When adopting the serial daisy chain, be sure to:

Meet the primary stage timing

As shown in Figure 2-41, the timing sent by the Host must meet the setup and hold time requirements for sampling D[0] from the rising and falling edges of CFG_CLK.

Meet the subsequent stage timing

The primary stage timing can be met by adjusting the Host output. The subsequent stage inputs all come from the previous stage, with adjustments mainly involving delays introduced by PCB traces, etc. As shown in the figure below, it is necessary to adjust delay parameters $T_{CLKDELAY}$ and $T_{DOUTDELAY}$ to meet the setup and hold time requirements of the subsequent stage devices for both edges.

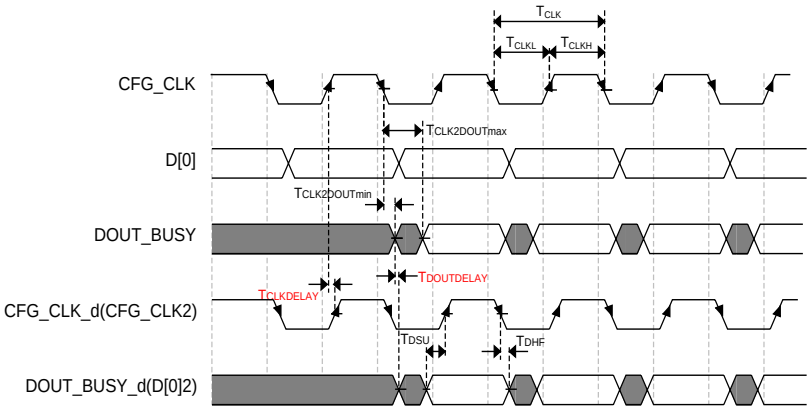

Figure 2-43 Timing Adjustment Example–Subsequent Stage

Note: For $T_{SSCLK2DOUT}$, T_{SSDSU}, and T_{SSDHF}, see Section 4.7 of the "***DS02001_Logos Family FPGAs Datasheet***".

The above figure shows that if the setup time for the rising edge is met when sending data on the falling edge, then the setup time for the falling edge will certainly be met; if the hold time for the falling edge is met, then the hold time for the rising edge will also be met. Therefore, $T_{CLKDELAY}$ and $T_{DOUTDELAY}$ must satisfy the following formula (unit: ns):

$$\begin{cases} T_{\mathrm{DSU}} = (T_{\mathrm{CLK}} + T_{\mathrm{CLKDELAY}}) - (T_{\mathrm{CLKH}} + T_{\mathrm{CLK2DOUTmax}} + T_{\mathrm{DOUTDELAY}}) \geq 2 \\ T_{\mathrm{DHF}} = T_{\mathrm{CLK2DOUTmin}} + T_{\mathrm{DOUTDELAY}} - T_{\mathrm{CLKDELAY}} \geq 1 \end{cases}$$

i.e.,

$$9.5 - T_{\mathrm{CLKL}} \leq T_{\mathrm{CLKDELAY}} - T_{\mathrm{DOUTDELAY}} \leq 1$$

2.6 Slave Parallel Configuration Mode

2.6.1 PGL12G/22G

The interface for the Slave Parallel configuration mode is shown in the figure below:

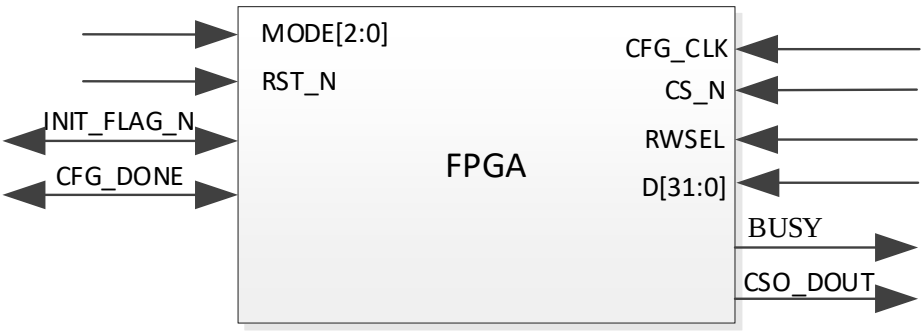

Figure 2-44 Slave Parallel Configuration Interface

The signals of the interface for the Slave Parallel configuration mode are described in the table below.

Table 2-13 Slave Parallel Configuration Interface Signal Description

Item	I/O	Dedicated/Multi-function	Description
RST_N	I	Dedicated	Asynchronous full-chip reset signal, active-low
CFG_CLK	I	Multi-funtion	Configures the clock
MODE[2:0]	I	Multi-funtion	Configuration mode pin 100: Slave Parallel mode
INIT_FLAG_N	open-drain	Multi-funtion	Before sampling pins MODE[2:0], INIT_FLAG_N is an input, and configuration can be delayed by keeping it at a low level. After sampling pins MODE[2:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
CFG_DONE	open-drain	Dedicated	Indicates configuration completion 0: FPGA not configured 1: FPGA configured
CS_N	I	Multi-funtion	External parallel interface chip select signal, active-low. Sampling data from the rising edge of CFG_CLK. (2)
RWSEL	I	Multi-funtion	External parallel interface read/write control signal. Sampling data from the rising edge of CFG_CLK. 0: Write 1: Read
D[31:0]	IO	Multi-funtion	Parallel data bus, sampling data from the rising edge of CFG_CLK and transmitting data
BUSY	O	Multi-funtion	Indicates whether the read-back data is ready 0: Ready

Item	I/O	Dedicated/Multi-function	Description
			1: Not ready
CSO_DOUT	O	Multi-funtion	Daisy chain data output, transmitted on the falling edge of CFG_CLK

Note:

1. The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.
2. CS_N is a synchronous signal. After INIT_FLAG_N is pulled high, before configuring the chip, CS_N must be kept high and provided at least 8 clock cycles for CFG_CLK to ensure the interface is in an initial operating state.

In the Slave Parallel mode, the power-up, start, and data loading of multiple chips on the board can be controlled via a master control chip (Host). The Host can be a microprocessor, CPLD, or another FPGA. To select the Slave Parallel mode, it is recommended to connect MODE [1] and MODE [0] to ground via pull-down resistors, and connect MODE [2] to the VCCIOL0 power supply via a pull-up resistor. The application is shown in the figure below:

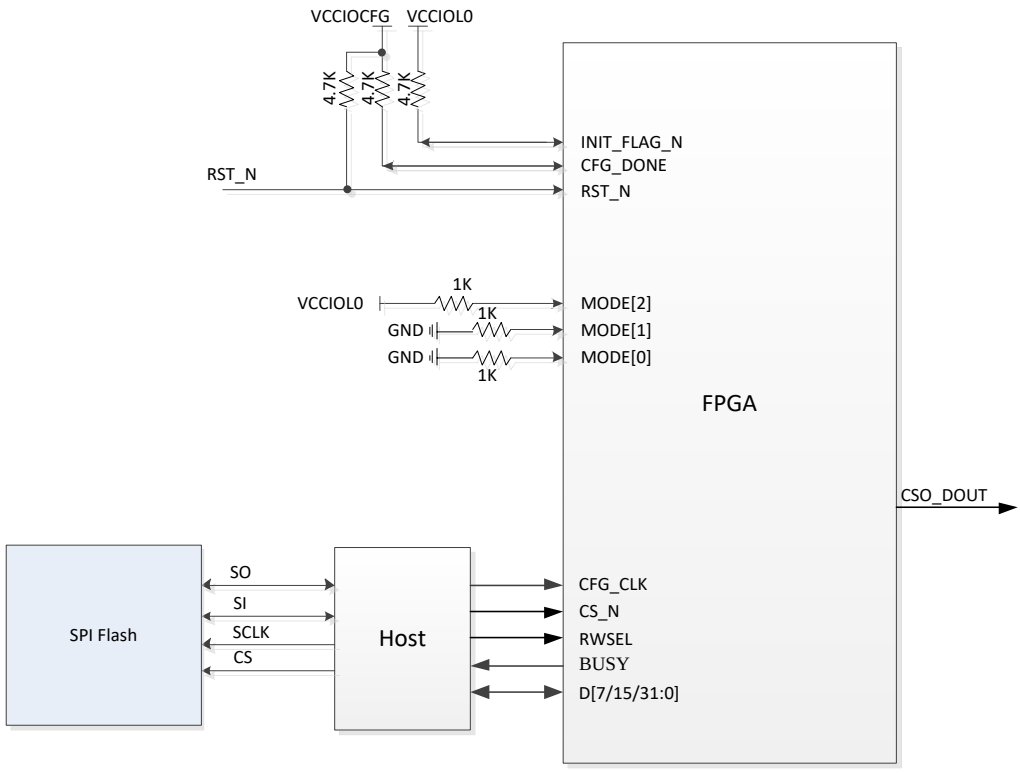

Figure 2-45 Application Diagram of the Slave Parallel Mode

In this mode, programming can be initiated by powering up or applying a low pulse to CONFIG_N; Monitor pins INIT_FLAG_N and CONFIG_DONE to judge whether programming is completed; When the Host sends a bitstream to the device, if wait for PLL Lock is not enabled (disabled by default in software), the device will release control of CFG_DONE during a clock cycle between the 100 NOP type 1 packet headers at the end of the bitstream, pulling CFG_DONE high via the

external pull-up resistor. The bitstream portion after CFG_DONE is pulled high provides a clock for waking up the device, so it is necessary to ensure the bitstream is completely transmitted before pulling CS_N high. If wait is enabled, the clock cannot be terminated before CFG_DONE is pulled high, and CFG_DONE should be pulled high after CFG_DONE is pulled high. After pulling CS_N high, at least 100 clocks must be provided continuously.

The Slave Parallel mode supports 8/16/32-bit data width modes, which can be selected by analyzing the bitstream. For details, see Bus Bit Width Auto Detection. When configuring with the Slave Parallel interface, pay attention to the correspondence between the data bus and the data bit order. Refer to

Bit Orders under Different Interface Bit Widths in Chapter 3.

The typical timing for the Slave Parallel configuration interface is shown in the figure below.

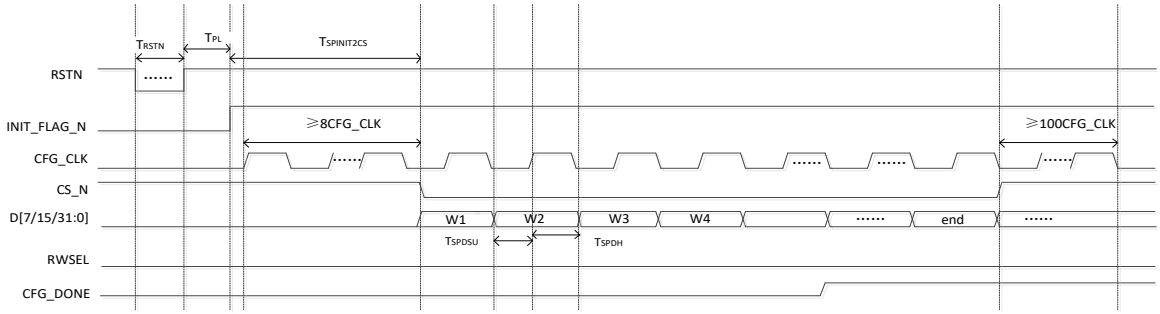

Figure 2-46 Slave Parallel Configuration Timing

Note: For T_{RSTN}, T_{PL}, $T_{SPINIT2CS}$, T_{SPDSU}, and T_{SPDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

When the Host cannot continuously send the bitstream, discontinuous loading can be achieved by controlling the CS_N or CFG_CLK of the Slave Parallel configuration interface, as shown in the figure below.

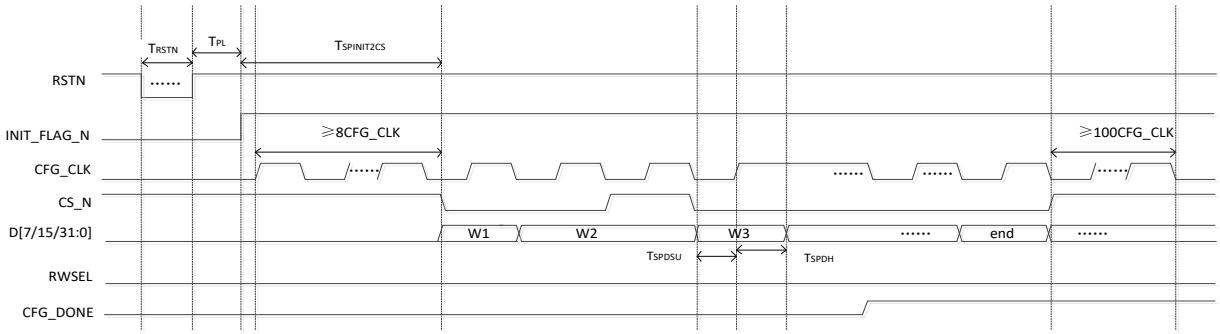

Figure 2-47 Slave Parallel Discontinuous Configuration Timing

Note: For T_{RSTN}, T_{PL}, $T_{SPINIT2CS}$, T_{SPDSU}, and T_{SPDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

After sending W2, data loading is paused by pulling CS_N high. It is important to maintain the data unchanged, that is, D[31:0] should remain unchanged while CS_N is high to avoid loading failure. After sending W3, data loading can also be paused by stopping the toggling of CFG_CLK, in which case there is no need to keep D[31:0] unchanged.

Additionally, the Slave Parallel interface also supports readback, with the typical timing shown in the figure below. For the specific readback process, refer to Slave Parallel Interface Readback.

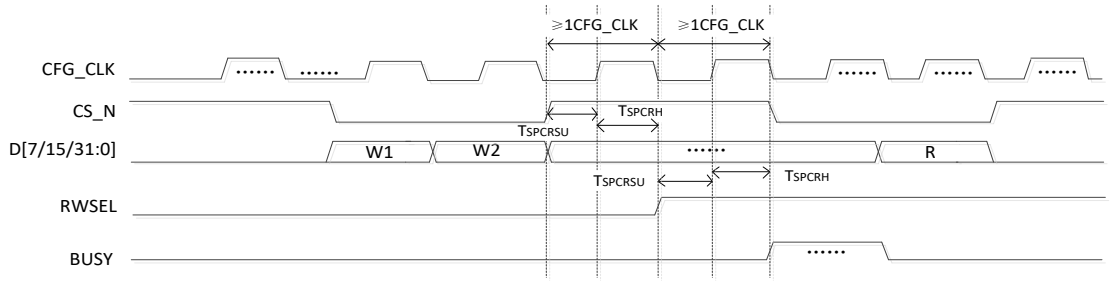

Figure 2-48 Slave Parallel Readback Timing

Note: For T_{SPCRSU} and T_{SPCRH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***". Slave Parallel Daisy Chain

Parallel daisy chain is a typical application of the Slave Parallel configuration mode. All devices can be configured in a parallel cascade using the Slave Parallel mode. Configure from the last (furthest) device to the first one in the chain during cascade configuration. Combine several bitstreams using the Fabric Configuration tool into a cascaded bitstream.

Through parallel cascade, multiple cascaded chips on the board can be configured via a master control chip (Host). Connect them as shown in the figure below.

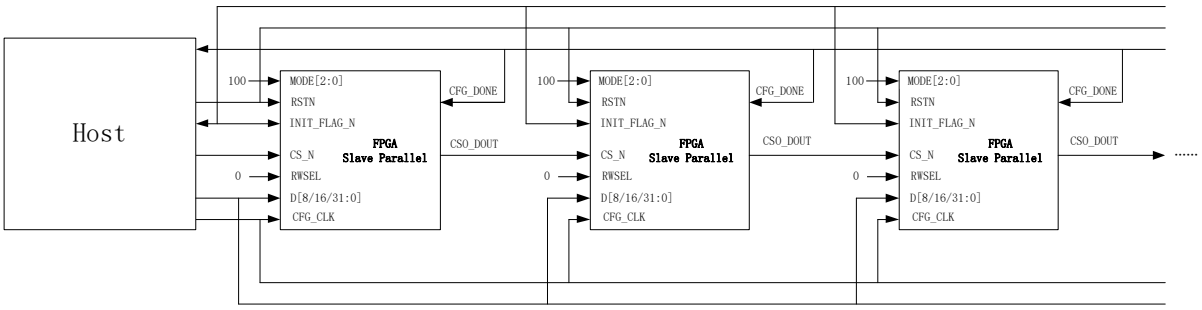

Figure 2-49 Daisy Chain Connection in the Slave Parallel Configuration Mode

2.6.2 PGL25G/50G/50H/100H

The interface for the Slave Parallel configuration mode is shown in the figure below:

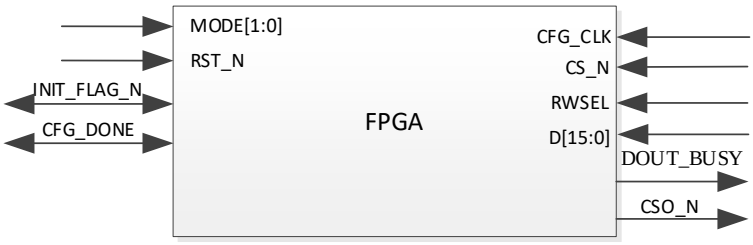

Figure 2-50 Slave Parallel Configuration Interface

The signals of the interface for the Slave Parallel configuration mode are described in the table below.

Table 2-14 Slave Parallel Configuration Interface Signal Description

Item	Property	I/O	Description
RST_N	Dedicated	I	Asynchronous full-chip reset signal, active-low
CFG_CLK	Multi-funtion	I	Configures the clock
MODE[1:0]	Multi-funtion	I	Configuration mode pin 10: Slave Parallel mode
INIT_FLAG_N	Multi-funtion	open-drain	Before sampling the MODE[1:0] pins, INIT_FLAG_N is an input, which can delay configuration by maintaining a low level. After sampling pins MODE[1:0], INIT_FLAG_N is open-drain, indicating whether an error occurred in the configuration process. 0: Wrong 1: Correct After the FPGA enters the user mode, if the readback CRC indication is enabled, this pin continues to be used as a configuration pin. If the readback CRC indication is disabled, this pin is released for users. (1)
CFG_DONE	Dedicated	open-drain	Indicates configuration completion 0: FPGA not configured 1: FPGA configured
CS_N	Multi-funtion	I	External parallel interface chip select signal, active-low. Sampling data from the rising edge of CFG_CLK. (2)
RWSEL	Multi-funtion	I	External parallel interface read/write control signal. Sampling data from the rising edge of CFG_CLK. 0: Write 1: Read
D[15:0]	Multi-funtion	IO	Parallel data bus, sampling data from the rising edge of CFG_CLK and transmitting data
DOUT_BUSY	Multi-funtion	O	Indicates whether the read-back data is ready 0: Ready 1: Not ready
CSO_N	Multi-funtion	O	Daisy chain data output, transmitted on the falling edge of CFG_CLK

Note:

1 The readback CRC indication can be enabled by software. Enabling the indication does not affect the functionality of the readback CRC: users can still use the readback CRC and view the results through the status register or internal parallel interface.

2 CS_N is a synchronous signal. After INIT_FLAG_N is pulled high, before configuring the chip, CS_N must be kept high and provided at least 8 clock cycles for CFG_CLK to ensure the interface is in an initial operating state.

In the Slave Parallel mode, the power-up, start, and data loading of multiple chips on the board can be controlled via a master control chip (Host). The Host can be a microprocessor, CPLD, or another FPGA. To select Slave Parallel mode, it is recommended to connect MODE[0] to ground through a pull-down resistor and MODE[1] to the VCCIO2 power supply through a pull-up resistor. The application is shown in the figure below:

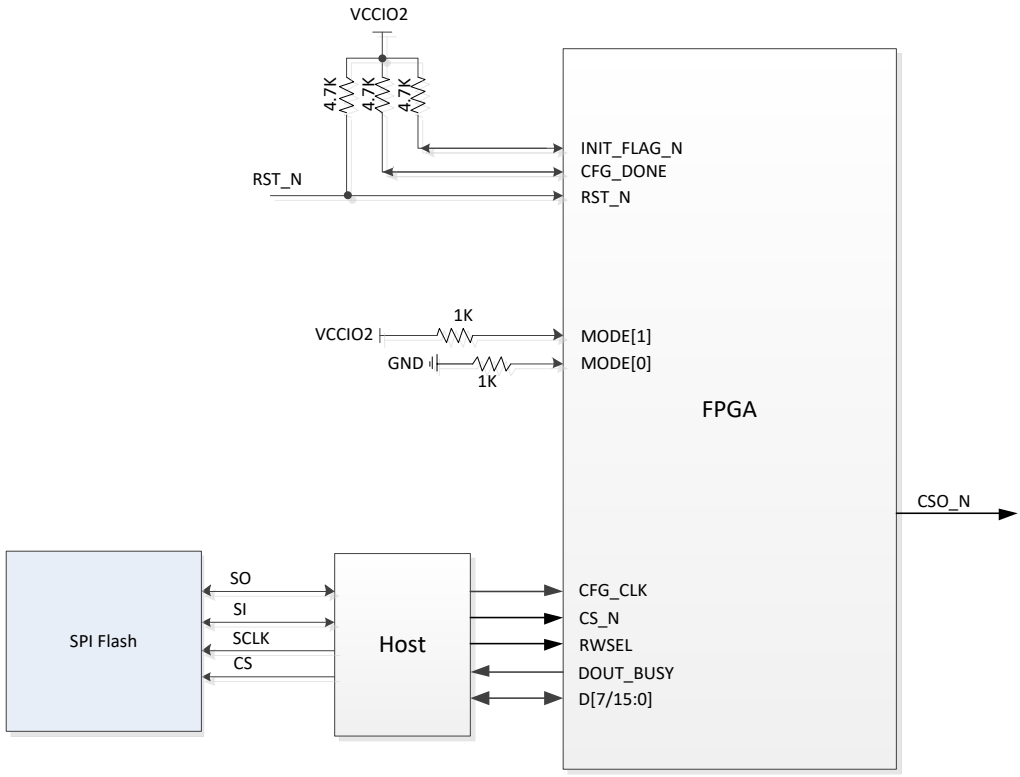

Figure 2-51 Application Diagram of the Slave Parallel Mode

In the Slave Parallel mode, multiple chips on the board can be powered up and started via a master control chip (Host);

In this mode, programming can be initiated by powering up or applying a low pulse to CONFIG_N;

Monitor pins INIT_FLAG_N and CONFIG_DONE to judge whether programming is completed;

When the Host sends a bitstream to the device, if wait for PLL Lock is not enabled (disabled by default in software), the device will release control of CFG_DONE during a clock cycle between the 100 NOP type 1 packet headers at the end of the bitstream, pulling CFG_DONE high via the external pull-up resistor. The bitstream portion after CFG_DONE is pulled high provides a clock for

waking up the device, so it is necessary to ensure the bitstream is completely transmitted before pulling CS_N high. If wait is enabled, the clock cannot be terminated before CFG_DONE is pulled high, and CFG_DONE should be pulled high after CFG_DONE is pulled high. After pulling CS_N high, at least 100 clocks must be provided continuously.

The Slave Parallel mode supports 8/16-bit data width modes, which can be selected by analyzing the bitstream. For details, see Bus Bit Width Auto Detection. When configuring with the Slave Parallel interface, pay attention to the correspondence between the data bus and the data bit order. Refer to Bit Orders under Different Interface Bit Widths in Chapter 3.

The typical timing for the Slave Parallel configuration interface is shown in the figure below.

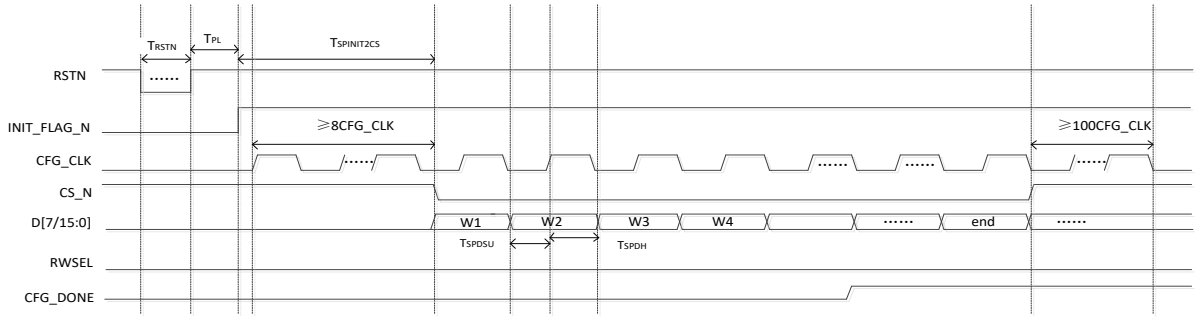

Figure 2-52 Slave Parallel Configuration Timing

Note: For T_{RSTN}, T_{PL}, $T_{SPINIT2CS}$, T_{SPDSU}, and T_{SPDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

When the Host cannot continuously send the bitstream, discontinuous loading can be achieved by controlling the CS_N or CFG_CLK of the Slave Parallel configuration interface, as shown in the figure below.

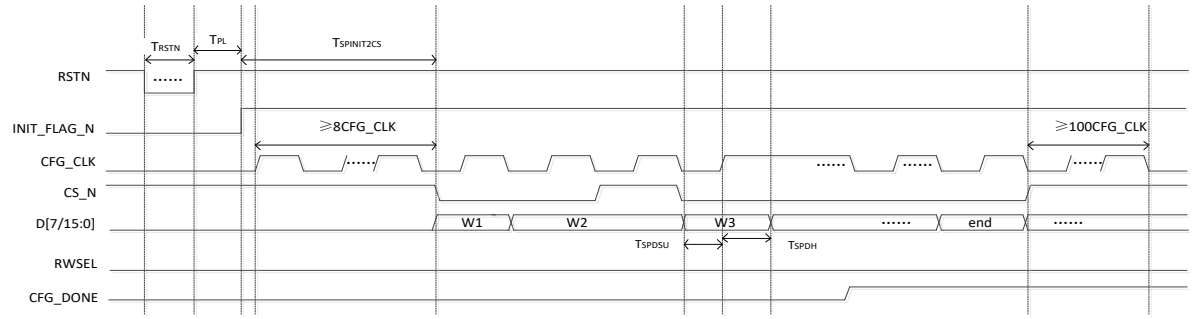

Figure 2-53 Slave Parallel Discontinuous Configuration Timing

Note: For T_{RSTN}, T_{PL}, $T_{SPINIT2CS}$, T_{SPDSU}, and T_{SPDH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

After sending W2, data loading is paused by pulling CS_N high. It is important to maintain the data unchanged, that is, D[15:0] should remain unchanged while CS_N is high to avoid loading failure. After sending W3, data loading can also be paused by stopping the toggling of CFG_CLK, in which case there is no need to keep D[15:0] unchanged.

Additionally, the Slave Parallel interface also supports readback, with the typical timing shown in the figure below. For the specific readback process, refer to Slave Parallel Interface Readback.

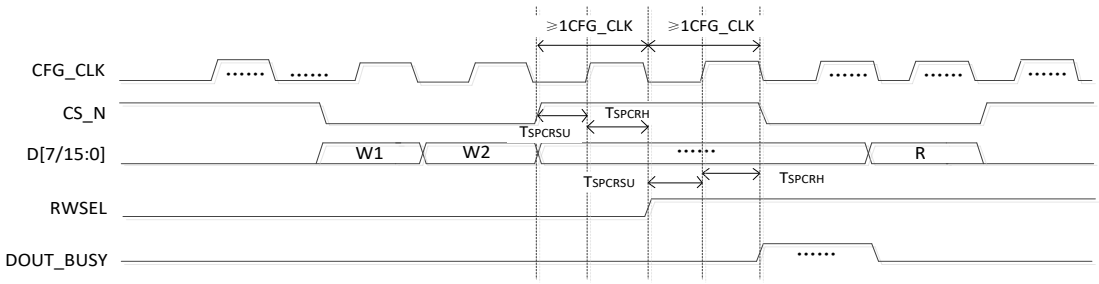

Figure 2-54 Slave Parallel Readback Timing

Note: For T_{SPCRSU} and T_{SPCRH}, see Section 4.7 of the "***DS02001 Logos Family FPGAs Datasheet***".

2.6.2.1 Slave Parallel Daisy Chain

Parallel daisy chain is a typical application of the Slave Parallel configuration mode. All devices can be configured in a parallel cascade using the Slave Parallel mode. Configure from the last (furthest) device to the first one in the chain during cascade configuration. Combine several bitstreams using the Fabric Configuration tool into a cascaded bitstream.

Through parallel cascade, multiple cascaded chips on the board can be configured via a master control chip (Host). Connect them as shown in the figure below.

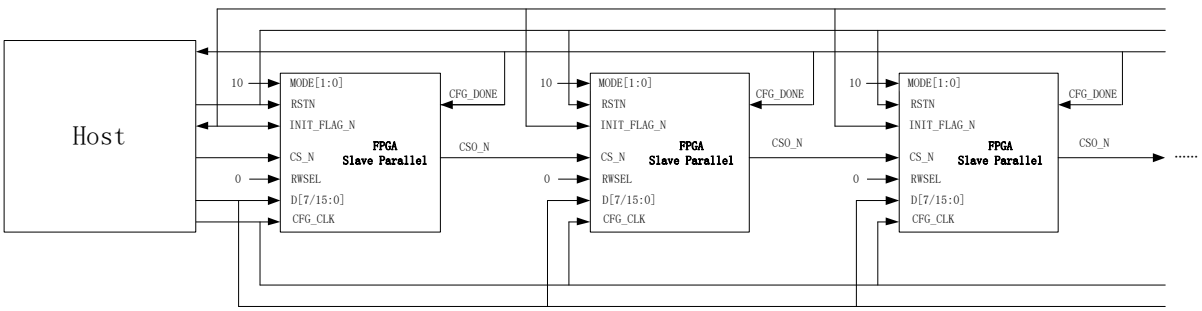

Figure 2-55 Daisy Chain Connection in the Slave Parallel Configuration Mode

Chapter 3 Download and Configuration

3.1 Configuration Process

The following figure shows the download and configuration process for the Logos Family FPGAs:

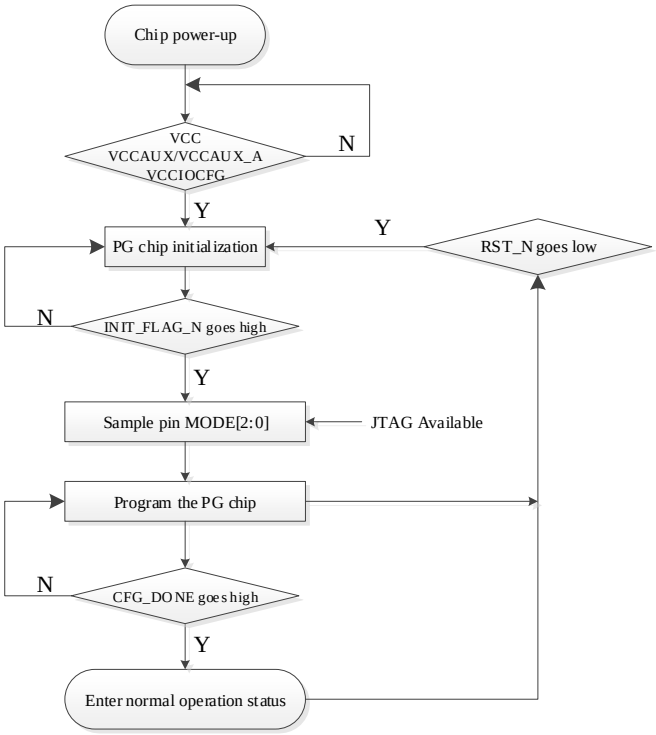

Figure 3-1 Download and Configuration Flowchart

For all configuration interfaces, the basic configuration steps are the same, including setup, bitstream loading, and wakeup.

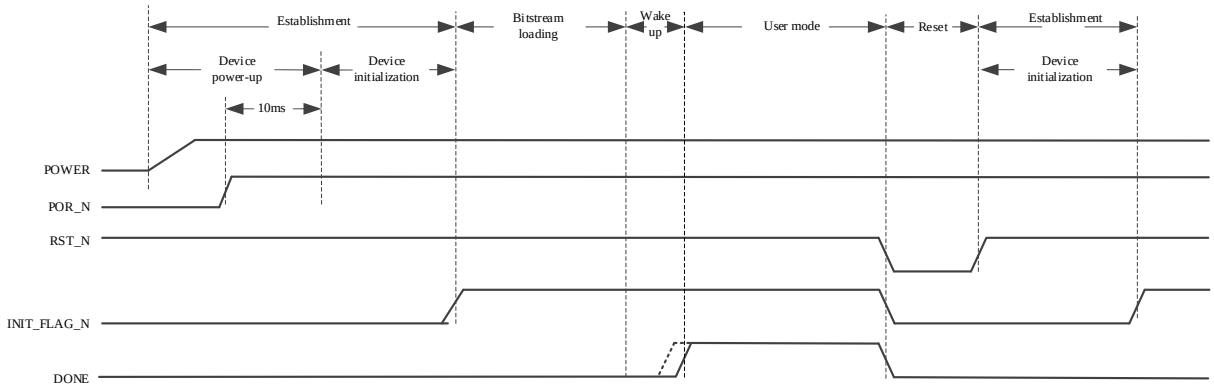

Figure 3-2 Download and Configuration Timing

The configuration process for the Logos Family FPGAs includes the following stages.

3.1.1 Setup

In the setup stage, the operations for FPGA device initialization and configuration mode determination should be completed, including device power-up, device initialization, and configuration mode selection.

3.1.1.1 Device Power-Up

The device power-up process is indicated by signal POR_N. When signal POR_N transitions from low to high, it indicates that power supply VCC of the configuration control system has risen to the device's operational threshold. After the device is powered up, the power-up reset circuit part of the CCS begins to operate, while the other circuit parts remain in the reset state. The power-up reset circuit counts using the main clock CLK of the CCS, which is generated by the device's internal crystal oscillator. Once the clock generated by the internal crystal oscillator stabilizes, the power-up reset of the CCS is released, and the CCS starts to operate.

3.1.1.2 Device Initialization

After the device is powered up and the CCS reset is released, the CCS starts to operate, performing device initialization.

In the device initialization stage, two operations are completed: clearing the configuration memory twice and reading the eFuse content. These two operations occur simultaneously, and the time it takes to clear the configuration memory twice is longer than that to read the eFuse content.

After entering the device initialization process, the CCS starts to clear the configuration memory frame by frame from address 0. After that, it performs the operation again. After the configuration memory is cleared twice, the initialization complete signal init_complete goes high.

The CCS can be kept in the device initialization process by maintaining the external INIT_FLAG_N pin low.

The CCS enters the device initialization process when any of the following functions is triggered: hard reset, JTAG instruction reset, warmboot, or version fallback.

3.1.1.3 Configuration Mode Selection

After the INIT_FLAG_N signal goes high, configuration mode pins MODE[2:0] are sampled to select the device's configuration mode. After mode sampling, the CCS selects the configuration function of the multi-function pins.

3.1.2 Bitstream Loading

Different configuration modes have different configuration interfaces. After configuration mode selection, the pins corresponding to the modes are set as configuration pins. This stage includes operations such as bus bit width check, synchronization, device ID check, pre-load option settings, data loading, CRC, and post-load option settings.

3.1.2.1 Bus Bit Width Auto Detection

In the Slave Parallel configuration mode, the data bus bit width is automatically detected by the CCS, which performs bit width matching by detecting the two-word bus bit width auto detection sequence 000000AA 08100020.

The bus bit width auto detection sequence is at the beginning of the bitstream, before the synchronization word. The CCS only detects the lower 8 bits of the parallel data bus. When the CCS performs detection, it first checks for the 0xAA byte on the lower 8 bits of the data bus. After detecting 0xAA, it continues to check whether the next byte is 0x08, 0x10, or 0x20, corresponding to 8-bit, 16-bit, or 32-bit data bus. If the next byte is not one of the aforementioned three, the CCS will re-detect 0xAA until it detects the sequence 0xAA + 0x08/0x10/0x20. Afterwards, the CCS sets the relevant data pins as configuration data pins, switches the data bus to the detected bit width, and begins searching for the synchronization word.

8-bit bus

FF	Padding word
FF	
FF	
FF	
00	Bus bit width auto detection sequence
00	
00	
AA	
08	
10	
00	
20	
FF	Padding word
FF	
FF	

FF	
...	

16-bit bus

FFFF	Padding word
FFFF	
0000	Bus bit width auto detection sequence
00AA	
0810	
0020	
FFFF	Padding word
FFFF	
...	

32-bit bus

FFFFFFFF	Padding word
000000AA	Bus bit width auto detection sequence
08100020	
FFFFFFFF	Padding word
...	

3.1.2.2 Synchronization

The synchronization word 0x01332D94 is used for 32-bit word boundary alignment. Only after synchronization can the subsequent deframing operations be performed.

Only when the correct synchronization word is received does the CCS consider the data to be valid; the CCS's packet processor then performs the unpacking operation; any data before the synchronization word, other than the bus width auto detection sequence, will be ignored.

3.1.2.3 Master Mode Read Operation Selection

In the Master SPI/Master BPI configuration mode, select the read operation.

In the Master SPI configuration mode, after selection, the device sends a read command to the SPI Flash again and readjusts the data bus bit width according to the read command.

In the Master BPI configuration mode, after selection, restart the asynchronous read timing if it is an asynchronous read command. If it is a synchronous read, first start the write BPI Flash read configuration register timing, set the BPI Flash to the synchronous read mode, and then start the

synchronous read timing.

3.1.2.4 Reset CRC

Reset the CRC register.

3.1.2.5 Device ID Check

Check whether the device ID in the bitstream matches the hardware; if not, the init_complete signal goes low, the ID error flag is stored in the bit id_err of the status register, and the configuration process is exited.

3.1.2.6 Pre-load Options Settings

Set pre-load options in the software, and specify the operating state of the device in the configuration register before loading configuration data:

Watchdog Settings

Decryption initial vector settings

Key selection

Decryption selection

Set whether to retain the multi-function configuration port for continued use as a configuration port after configuration is completed

Wakeup clock selection

Wakeup timing settings

Whether to wait for PLL upon wakeup

Version fallback settings

Master mode frequency settings

3.1.2.7 Load Configuration Data

Write the configuration data frames into the configuration memory.

3.1.2.8 Post-load CRC

After the configuration data is loaded, perform a CRC. The CRC value generated by the software is stored in the CRC register and compared with the CRC value calculated by the CCS. If the two

values match, it indicates the CRC has passed and all configuration information has been correctly written into the configuration memory. If the two values do not match, it indicates the CRC has failed, causing the INIT_N signal to go low and the CRC error flag to be stored in bit crc_err of the status register.

3.1.2.9 Post-load Option Settings

After the configuration data is loaded, specify the operating state of the device in the configuration register:
Set whether to allow user logic to turn off the OSC
External port security level settings

3.1.3 Wakeup

After the bitstream is loaded and passes the CRC, the FPGA enters the wakeup stage. The configuration system first enables the logical outputs of all functional modules inside the FPGA. Before wakeup, a CRC is performed again followed by a desynchronization operation, indicating the end of configuration, and subsequent operations require resynchronization. Finally, the wakeup is completed, and the corresponding global signals are released gradually.

3.1.3.1 Enables Global Logics

After loading, once the CRC is passed, the logical outputs of all functional modules within the chip are enabled.

3.1.3.2 Start to Wakeup

The wakeup operation is initiated after global logic is enabled.

3.1.3.3 Pre-wakeup CRC

A CRC is performed before executing the wakeup operation. The CRC value generated by the software is stored in the CRC register and compared with the CRC value calculated by the CCS. If the two values match, it indicates the CRC has passed and the pre-wakeup preparation is completed. If the two values do not match, it indicates the CRC has failed, causing the INIT_N signal to go low and the CRC error flag to be stored in bit crc_err of the status register.

3.1.3.4 Desynchronization

This indicates the end of the configuration process, requiring resynchronization for subsequent operations.

3.1.3.5 Wakeup

After desynchronization is completed, the wakeup circuit begins to operate according to the set timing, gradually releasing global signals. The wakeup timing can be set in PDS (Project->Project Setting->Generate Bitstream->Startup). The following introduces each global signal (based on this wake-up sequence: DONE-T1, GOUTEN-T2, GWEN-T3, GRS_N-T4, with the default wake-up sequence specified in PDS).

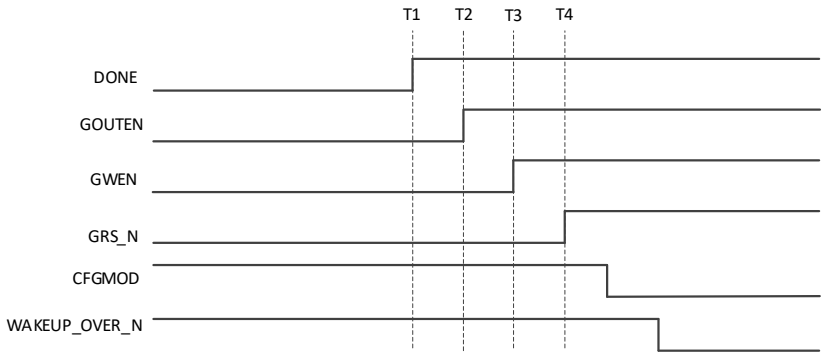

Figure 3-3 Wakeup Timing

The global logic enable signal GLOGEN is output to the fabric, enabling all logic blocks; it is then fed back to the CCS from the farthest end. After the CCS receives the feedback signal GLOGEN_FB, it enters the wakeup timing. Before GLOGEN is enabled, the register output values of all modules in the chip are 1; After it is enabled, the module registers are controlled by GRS_N.

Before configuration is completed, the global register set/reset signal GRS_N is at a low level, controlling the GRS_N ports of all modules in the chip, putting the registers of modules in a set/reset state. Whether the registers in user logic are in a set or reset state depends on the user design and software. After configuration is completed, the GRS_N signals of all modules in the chip are released, putting the registers of modules to a non-global set/reset state.

Before configuration is completed, the global IO output enable signal GOUTEN is at a low level, and all IO outputs are high-impedance. After configuration is completed, all IO outputs are enabled.

Table 3-1 GOUTEN Logical Functions

GOUTEN	**0**	**1**
IO used by users	High-Z	User logic control
I/O unused by users	High-Z	Bitstream settings (1)

Note:

1. This can be set in the PDS tool: Project->Project Setting->Generate Bitstream->General->Unused IO Status.

The global write enable signal GWEN can enable write for the global internal storage resources of the FPGA chip. Before configuration is completed, this signal is at a low level, disabling the write operation for the FPGA's storage resources.

The multi-function pin control signal CFGMOD releases the multi-function pins used during the configuration process for users.

The wakeup completion indicator signal WAKEUP_OVER_N indicates the successful completion of the wakeup process.

Users can configure the time when global wakeup signals GOUTEN, GRS_N, GWEN, and CFG_DONE are set to 1, the configurable options include T1, T2, T3, or T4.

The number of cycles between two times can be set by users.

After wakeup, the FPGA transitions from the configuration mode to the user mode.

To reload bitstreams, users can pull RST_N low for a period of time and then release it, and the FPGA will repeat the above process.

Delayed Wakeup

The global wakeup signals GOUTEN, GRS_N, GWEN, and DONE can be set to 1 by users at time T1, T2, T3, or T4.

The number of cycles between two times can be set by users.

The configuration completion indicator signal DONE drives pin CFG_DONE. After CFG_DONE is set to 1, the drive control of the pin is released. Pin CFG_DONE is an open-drain output pin and usually connected to an external pull-up resistor. If the DONE signal goes high and the CFG_DONE pin does not go high (external drive is low), then the wakeup process will be paused and not resumed until the CFG_DONE pin goes high.

After DONE is set to 1, subsequent operations can be carried out only after the external drive CFG_DONE is high. Signals that are set to 1 after DONE is set to 1 need to wait for CFG_DONE to be high. Signals set to 1 before DONE is set to 1 or concurrently with DONE are not affected by the external drive CFG_DONE.

For example, set GOUTEN to 1 at T1, DONE to 1 at T2, GWEN to 1 at T3, and GRSN to 1 at T4. Then GOUTEN, DONE are set to 1 according to the set timing. At time T3, if the external drive CFG_DONE is 1, then GWEN is set to 1; otherwise, GWEN is delayed by a time slot (the number of cycles between two times is settable), and CFG_DONE can be checked again in the next time slot. If CFG_DONE is high, then GWEN is set to 1; otherwise, it is delayed by another time slot. The time when GRSN is set to 1 is delayed sequentially based on the time when GWEN is set to 1. The delay wakeup can be set in PDS: Project->Project Setting->Generate Bitstream->Startup.

3.2 Configuration Files

3.2.1 Bitstream Generation

The PDS design software generates configuration files with different extensions to accommodate different configuration schemes, as detailed in the table below.

Table 3-2 Descriptions of PDS Configuration Files

Configuration File Extension	Description
.sbit	Binary configuration data that includes header information (bitstream name, date, etc.). The configuration tool recognizes the header information but does not write it into the FPGA. Files with the extension of .sbit can be written directly into the FPGA using the Fabric Configuration tool through Cable.
.bin	Binary configuration data without header information (pure bitstream), suitable for user configuration schemes, such as microprocessor-based FPGA configuration.
.sfc	The bitstream file written into Flash, which is converted from the .sbit file.

Files with the extension of .sbit are default bitstream files generated after compilation by the PDS tool, as shown below. Running Generate Bitstream will produce files with .sbit.

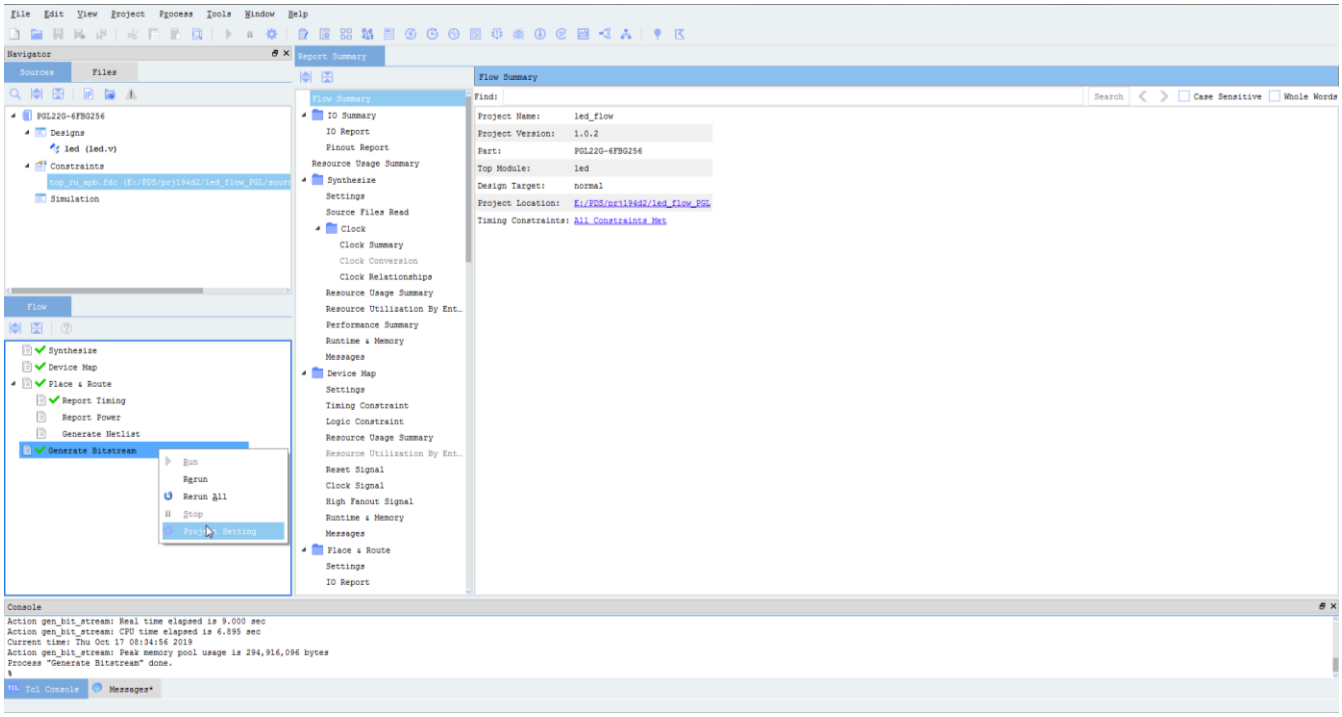

Figure 3-4 Bitstream Generation Diagram

As shown in the above figure, right-click on [Generate Bitstream], select configure, and a window as shown below will pop up. Check the [Create Bin File] option, click [OK], and rerun [Generate Bitstream] to generate files with both .sbit and .bin simultaneously.

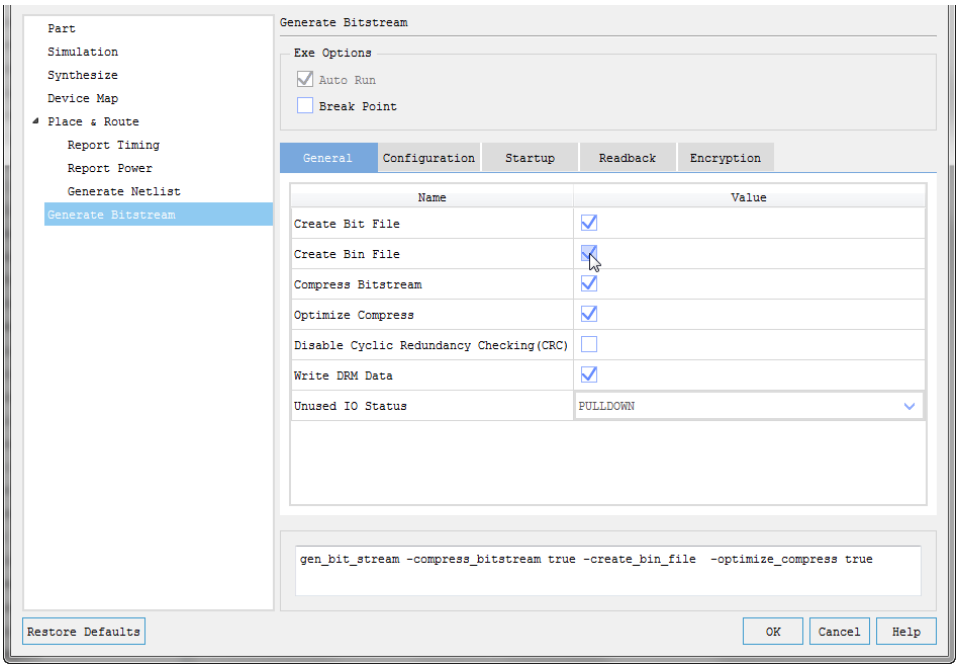

Figure 3-5 Configure Generate Bitstream

In the [Fabric Configuration] tool interface, click on [Operations] in the menu bar, and select [Generate Flash Programming File] from the dropdown menu to display the interface as shown below.

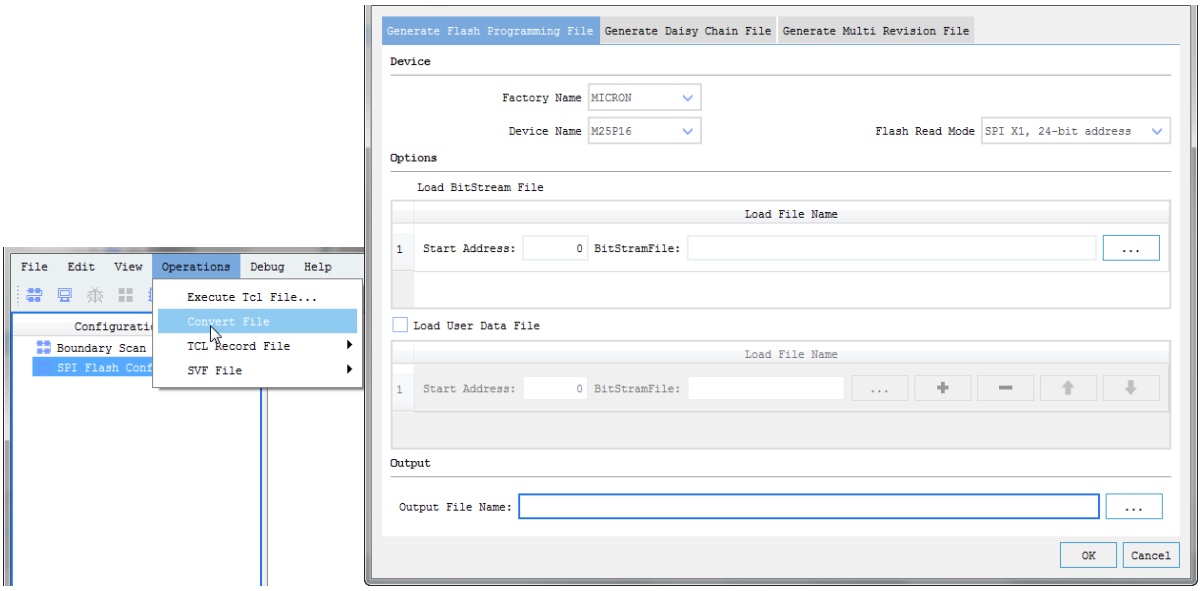

Figure 3-6 Generate Flash Programming File

Users should select the appropriate Flash Device based on their actual application, choose the bit width and address bit width in the [Flash Read Mode] dropdown box, then enter the path for the sbit file and the storage path for the sfc file, and click ok to generate the Flash configuration file—the sfc file.

3.2.2 Bitstream Size

The bitstream sizes for the Logos Family devices are shown in the table below. The compressed bitstream files are smaller than those before compression. The size of the compressed files depends on the design, which is not explained here.

Table 3-3 Bitstream Sizes of Logos Family Devices

Device	**File Name**	**Uncompressed File Size (Kbyte)**
PGL12G	*.sbit	518
PGL22G	*.sbit	745
PGL22GS	*.sbit	745
PGL25G	*.sbit	982
PGL50G	*.sbit	2052
PGL50H	*.sbit	2052
PGL100H	*.sbit	4059

3.2.3 Bit Orders under Different Interface Bit Widths

The Logos Family FPGAs configuration supports multiple bit widths. Different configuration modes with the same bit width have a consistent bit order, i.e., the correspondence between the high and low bits of the data bus and the bitstream sequence are consistent. For example, the bit order for Master SPI (x8) and Master BPI (x8) modes is the same. Taking the sync word (32'h01332D94) in the bitstream as an example, the high and low bits of the data bus correspond to the bitstream sequence, as shown in the table below.

Table 3-4 Sync Word Transmission Sequence under Different Bit Widths

Clock Cycle	**Numeral System**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
D[1:0](x2)	2'b	00	00	00	01	00	11	00	11	00	10	11	01	10	01	01	00
D[3:0](x4)	4'h	0	1	3	3	2	D	9	4								
D[7:0](x8)	8'h	01	33	2D	94												
D[15:0](x16)	16'h	0133	2D94														
D[31:0](x32)	32'h	01332D94															

Note: In the table, D[x:0] only represents the high and low bits of the data bus. For specific corresponding pins, refer to the list of ports under different interface modes.

3.3 Configuration Data Packets

3.3.1 Packet Types

There are two types of packets for the Logos Family FPGAs configuration bitstreams: Type 1 and Type 2.

Type 1 packets are used for register read and write operations. A type 1 packet consists of a header and data.

The header of a type 1 packet is a 32-bit word. Data follows the header. Data unit is 32-bit word.

If the number of 32-bit words following the header is 0, then the packet is empty with header and without data.

The format of the type 1 packet header is shown in the table below.

Table 3-5 Type 1 Packet Header Format

[31:29]	**[28:27]**	**[26:22]**	**[21:0]**
Packet Header Type	Op Code	Register address	32-bit word count
101	00: No operation; 01: Write; 10: Read; 11: Reserved	Register address	The number of 32-bit words following the header

Type 2 packets are used for reading and writing to the configuration memory. A type 2 packet consists of a header and data.

The header of a type 2 packet is a 32-bit word. Data follows the header. Data unit is 32-bit word.

Type 2 packets must follow a type 1 empty packet. Type 2 packets do not specify a register address, but follow the address specified by the preceding type 1 empty packet.

The format of the type 2 packet header is shown in the table below.

Table 3-6 Type 2 Packet Header Format

[31:29]	**[28:27]**	**[26:0]**
Packet Header Type	Op Code	32-bit word count
010	00: No operation; 01: Write; 10: Read; 11: Reserved	The number of 32-bit words following the header

3.3.2 Configuration register

The configuration registers for the Logos Family FPGAs are listed in the table below.

Table 3-7 Descriptions of Configuration Registers

Item	**R/W**	**Address**	**Description**
CRCR	R/W	00000	CRC Registers
IDR	R/W	00001	Device identification register
CMDR	R/W	00010	Command register
CTRL0R	R/W	00011	Control register 0
CTRL1R	R/W	00100	Control register 1
CMEMIR	W	00101	Frame data input register
MFWRITER	W	00110	Multi-frame write register
CMEMOR	R	00111	Frame data output register
IVR	W	01000	Decryption initial vector register
STATUSR	R	01001	Status register
CHAINR	W	01010	Cascade register
ADRR	R/W	01011	Frame address register
SBPIR	R/W	01100	SBPI register
IRSTCTRLR	R/W	01111	Warm boot control register
IRSTADRR	R/W	10000	Warm boot address register
WATCHDOGR	R/W	10001	Watchdog status register
HSTATUSR	R	10010	Historical status register
FADR0R	R/W	10011	Bitstream 0 address register
FADR1R	R/W	10100	Bitstream 1 address register
FADR2R	R/W	10101	Bitstream 2 address register
FADR3R	R/W	10110	Bitstream 3 address register
CMASKR	R/W	10111	Control master register

Item	R/W	Address	Description
FALLBACKR	R/W	11000	Version fallback register
OPTION0R	R/W	11001	Option register 0
OPTION1R	R/W	11010	Option register 1
RCRR	R/W	11011	RCR control register

The configuration registers are detailed as follows.

3.3.2.1 CRC Register (CRCR)

Each write to the CRC register corresponds to a CRC for the bitstream. If the value written to the CRC register matches the current CRC value calculated by the CCS, then the CRC passes. Otherwise, init_complete is pulled low.

3.3.2.2 Device ID Register (IDR)

Before loading the bitstream, write the IDCODE to the IDR. Subsequent operations can be performed only if the written IDCODE matches the device IDCODE. Each write to the ID register corresponds to a device ID check. If the lower 28 bits of the value written to the ID register matches that of the device's ID, then the ID check passes. Otherwise, init_complete is pulled low.
Device models are shown in the table below.

Table 3-8 Logos Family FPGAs Device Models

Device Model	Device ID
PGL12G	X0501899
PGL22G	X0303899
PGL22GS	X0303899
PGL25G	X0511899
PGL50G	X05A1899
PGL50H	X0521899
PGL100H	X0522899

3.3.2.3 Command Register (CMDR)

The command register determines the next operation to be performed by the CCS. Write at most one command to the command register per operation. A single packet header cannot be followed by multiple commands.

Table 3-9 CCS Commands and Their Descriptions

Command	Code	Description
NOP	00000	No Op
RSTCRC	00001	Reset CRC Reset CRC register
SWITCH	00010	Switch the master mode configuration clock mclk frequency Initiate the operation to update the master mode configuration clock mclk frequency, with the value determined by oscfsel in CTRL0R
SWITCHCLK	00011	Switch system clock clk frequency Initiate the operation to update the system clock clk frequency, with the value determined by oscfsel_clk in CTRL0R
WCMEM	00100	Write configuration data Used before writing configuration data into the configuration memory through CMEMIR
MFWRITE	00101	Multi-frame write Write a currently written frame of data into the subsequent consecutive frames. The number of frames is determined by MFWRITER
RCMEM	00110	Read configuration data Used before reading back configuration data from the configuration memory through CMEMOR
SWAKEUP	00111	Start wakeup operation
SWAKEDOWN	01000	Disable wakeup operation
GUP	01001	Enables internal logic
GDOWN	01010	Internal logic not enabled
DESYNC	01011	Desynchronization Used at the end of configuration
RWD	01100	Restart the watchdog
RRBCRC	01101	Reset readback CRC
RBCRC	01110	Readback CRC
IRST	01111	Warm boot Warm boot is invalid during version fallback

3.3.2.4 Control Register 0 (CTRL0R)

Table 3-10 Descriptions of CTRL0R

Bit	Item	Initial Value	Description
[31:5]	Reserved		
[4]	spersist	1’b0	Selects whether the Slave SPI interface is reserved in the user mode 0: Released for users to use 1: When the Slave Parallel interface is released for users in the user mode, the Slave SPI interface remains reserved for the configuration port. When the Slave Parallel interface remains reserved for the configuration port in the user mode, the Slave SPI interface is released for users.
[3]	Reserved		
[2]	tkey_en	1’b0	Enables a temporary key 0: Use the key 1: Use a temporary key

Bit	Item	Initial Value	Description
[1]	persist	1’b0	Selects whether the Slave Parallel interface is reserved in the user mode 0: Released for users to use 1: Reserved for continued use as a configuration port
[0]	dec_en	1’b0	Enables decryption

3.3.2.5 Control Register 1 (CTRL1R)

Table 3-11 Description of Control Register 1

Bit	Item	Initial Value	Description
[31:4]	Reserved		
[3]	mfg_por_off	1’b0	0: Normal monitoring of VCC, VDDM (internal LDO output voltage), VCCIOCFG, VCCAUX voltages 1: When testing SRAM retention voltage is required, with gwen is set to 1, turn off the fine probing function of VCC and VDDM. At this time, POR will reassert only when VCC<0.65V or VDDM<0.5V.
[2]	osc_off	1’b0	Controls whether user logic is allowed to shut down OSC 0: Not allow 1: Allow
[1:0]	wrctrl	2’b00	External port security level. Controls the shutdown of reconfiguration, partial reconfiguration, local dynamic reconfiguration, and readback configuration memory. Once configured to disabled, it cannot be changed back to enabled unless reset. 00: Enables reconfiguration and readback 01: Reconfiguration enabled, readback disabled 1x: Reconfiguration disabled, readback disabled

3.3.2.6 Control Mask Register (CMASKR)

The control mask register is used to mask corresponding bits in control register 0 and control register 1, with a default value of 32'd0. 0 for masking. For example, when the value of the control mask register is 32'h0000_0003, the lowest 2 bits of control register 0 and control register 1 can be written to, while the other bits cannot.

3.3.2.7 Option Register 0 (OPTION0R)

Table 312 Description of Option Register 0

Bit	Item	Initial Value	Description
[31:24]	Reserved		
[23]	done_syn	1’b0	Enables external done synchronization. Indicates whether the external input done is the synchronous signal for the output done. If it is a synchronous signal, the wakeup module of the CCS uses it directly. If it is an asynchronous signal, it needs to be synchronized before use, typically for the delayed done function. 0: Do not synchronize

Bit	Item	Initial Value	Description
			1: Synchronize
[22:21]	t_sel	2’d0	Wakeup cycle length selection, i.e., the time interval between T1, T2, and T3 2’b00: 1 2’b01: 2 2’b10: 4 2’b11: 1
[20:19]	done_sel	2’d0	Selects the time when the done signal is pulled high 2’d0: T3 2’d1: T1 2’d2: T2 2’d3: T4
[18:17]	gwen_sel	2’d0	Selects the time when the gwen signal is pulled high 2’d0: T2 2’d1: T1 2’d2: T3 2’d3: T4
[16:15]	grsn_sel	2’d0	Selects the time when the grsn signal is pulled high 2’d0: T2 2’d1: T1 2’d2: T3 2’d3: T4
[14:13]	gouten_sel	2’d0	Selects the time when the gouten signal is pulled high 2’d0: T1 2’d1: T2 2’d2: T3 2’d3: T4
[12]	Reserved	1’b0	Reserved
[11]	wait_pll	1’b0	Wait for PLL lock to be enabled during wakeup 0: No need to wait for PLL lock 1: Wait for PLL lock
[10:9]	startup_sel	2’b00	Wakeup clock selection 00: MCLK 01: SCLK 10: TCK 11: UCLK For the wakeup clock, only the clock used during configuration or the user clock can be selected
[8:6]	oscfsel_clk	3’b010	System clock CLK frequency selection 3'd2:100M 3'd3:66.67M 3'd4:50M 3'd5:40M 3'd6:33.33M 3'd0:25M
[5:0]	oscfsel	6’d0	Master mode configuration clock MCLK frequency selection 6'd2:100M 6'd3:66.67M 6'd4:50M 6'd5:40M 6'd6:33.33M 6'd8:25M 6'd10:20M 6'd16:12.5M 6'd20:10M 6'd25:8M

Bit	Item	Initial Value	Description
			6'd32:6.25M 6'd40:5M 6'd50:4M 6'd0:3.125M

3.3.2.8 Option Register 1 (OPTION1R)

Table 3-13 Selection Register 1

Bit	Item	Initial Value	Description
[31:2]	Reserved		
[1]	rbcrc_disable	1’b0	Disable readback CRC error indication 0: Enabled 1: Disabled
[0]	crc_disable	1’b0	Disable CRC 0: Enabled 1: Disabled

3.3.2.9 Frame Data Input Register (CMEMIR)

The CMEMIR is the configuration data interface for the configuration memory. Data is written into the configuration memory through the CMEMIR.

3.3.2.10 Multi-Frame Write Register (MFWRITER)

The number of compressed frames indicates that the data content of several consecutive frames following is the same as that of the just-written frame. For instance, after configuring a frame of data, if the data content of the next three consecutive frames is the same as that frame, then the MFWRITER should be written with a 3.

3.3.2.11 Frame Data Output Register (CMEMOR)

The CMEMOR is the readback data interface for the configuration memory. Data is read back from the configuration memory through the CMEMOR.

3.3.2.12 Decryption Initial Vector Register (IVR)

128-bit initial vector, used for decryption.

3.3.2.13 Status Register (STATUSR)

The status register is described as follows:

Table 3-14 Status Register Description

Bit	Item	Description
[31:30]	Reserved	
[29]	over_temp	Overtemperature flag
[28]	flg_x32	Indicates the 32-bit bit width in the Slave Parallel mode
[27]	flg_x16	Indicates the 16-bit bit width in the Slave Parallel mode
[26]	flg_x8	Indicates the 8-bit bit width in the Slave Parallel mode
[25:24]	ipal_m[1:0]	Selects the ipal data width
[23]	key_lock	Key lock flag
[22]	fallback	Fallback indication flag
[21]	Reserved	Reserved
[20]	pll_lock	PLL lock
[19]	gwen	GWEN
[18]	grsn	GRS_N
[17]	gouten	GOUTEN
[16]	vddt_n	vddt_n
[15]	glogen_fb	glogen feedback
[14]	glogen	glogen
[13]	done_i	External done
[12]	done	done
[11]	init_b	init_b
[10]	init_complete	Configuration completion and error indication
[9:7]	m[2:0]	Mode selection
[6]	wakedown_over	Wakeup shutdown over
[5]	wakeup_over	Wakeup over
[4]	overflow	FIFO overflow
[3]	timeout	Watchdog timeout
[2]	rbcrc_err	Readback CRC results 0: CRC correct 1: CRC error
[1]	crc_err	CRC results 0: CRC correct 1: CRC error
[0]	id_err	ID detection results 0: Correct 1: Wrong

3.3.2.14 Watchdog Register (WATCHDOGR)

The watchdog register is used for timeout detection in the configuration mode and user mode. It is described as follows.

Table 3-15 Watchdog Register Description

Bit	Item	Initial Value	Description
[31]	wd_user_en	1’b0	Enables the watchdog in the user mode
[30]	wd_cfg_en	1’b0	Enables the watchdog in the configuration mode
[29:0]	wd_value	30’h3FFF_FFFF	Watchdog timeout value

3.3.2.15 Frame Address Register (ADDR)

The frame address register provides the starting addresses for configuration and read-back data. It can transmit only valid addresses, rather than those beyond the range.

Table 3-16 Frame Address Register Description

Bit	Item	Initial Value	Description
[31:27]	Reserved		
[26:25]	type	2’d0	Configuration memory content type 00: Non-DRM content 01: DRM content 10: Reserved 11: Reserved
[24:20]	addr_region	5’d0	Region address
[19:18]	Reserved		
[17:10]	addr_column	8’d0	Column address
[9:8]	Reserved		
[7:0]	addr_frame	8’d0	Frame address

3.3.2.16 SBPI Register (SBPIR)

The SBPI register provides control and option information for the Master SPI mode, internal Master SPI mode, and Master BPI mode.

Table 3-17 SBPI Register Description

Bit	Item	Initial Value	Description
[31]	emclk_en	1’b0	External master clock enable 1'b0: The main mode configuration clock uses the internal clock 1'b1: The main mode clock uses the external clock
[30]	sbpi_rf_sel	1’b0	Data sampling clock edge selection 0: Rising edge 1: Falling edge In the Master SPI mode, sampling on the falling edge is in the fast mode

Bit	Item	Initial Value	Description
			In the Master BPI asynchronous mode, sampling on the falling edge is in the fast mode In the Master BPI synchronous mode, sampling on the rising edge is in the fast mode
[29:21]	Reserved		
[20:19]	bpi_cycle	2'd0	BPI Flash page start address clock cycle count 00:1 01:2 10:3 11:4
[18:17]	bpi_page	2'd0	BPI Flash page size 00: 1 byte/halfword 01: 4 bytes/halfwords 10: 8 bytes/halfwords 11: 16 bytes/halfwords
[16]	bpi_mode	1'b0	BPI Flash read mode selection 0: Asynchronous 1: Synchronous
[15:11]	Reserved		
[10]	addrwidth	1'b0	SPI address bit width 0: 24-bit address 1: 32-bit address
[9:8]	datawidth	2'd0	SPI bit width 00: x1 01: x2 10: x4 11: x8
[7:0]	opcode	8'h0B	SPI master mode operation code

3.3.2.17 RCR Control Register (RCRR)

The RCR register provides operation codes and data for programmers to read configuration register instructions in the Master BPI mode.

Table 3-18 RCR Control Register Description

Bit	Item	Initial Value	Description
[31:24]	confirm_prcr	8'h03	Operation codes of the write cycle confirmation for programmers to read configuration register instructions
[23:16]	setup_prcr	8'h60	Operation codes of the write cycle setup for programmers to read configuration register instructions
[15:0]	rcr	16'd0	Content of the read configuration register to be programmed into the BPI flash

3.3.2.18 Warm boot Control Register (IRSTCTRLR)

Warm boot control

Table 3-19 Descriptions of Warm boot Control Registers

Bit	Item	Initial Value	Description
[31:2]	Reserved		
[1]	vs_en	1’b0	vs[1:0] pin enable 0: Disabled 1: Enabled
[0]	vback_en	1’b0	Enables version fallback. Controls fallback to either the golden bitstream or the previous version bitstream 0: Disabled 1: Enabled

3.3.2.19 Warm boot address register (ADDR)

Flash start address during warm boot, byte addressing.

Table 3-20 Warm boot Address Register Description

Bit	Item	Initial Value	Description
[31:24]	Flash start address	8‘d0	Flash start address[31:24] for 4-byte mode
[23:0]	Flash start address	24’d0	Flash start address

3.3.2.20 Historical Status Register (HSTATUSR)

Table 3-21 Historical Status Register Description

Bit	Item	Description
[31:15]	Reserved	
[14]	timeout1	Previous state timeout 0: Watchdog not timed out 1: Watchdog timeout
[13]	rbcrc_err1	Previous state readback CRC error 0: Correct 1: Wrong
[12]	crc_err1	CRC error of the last status 0: Correct 1: Wrong
[11]	id_err1	Previous state ID error 0: Correct 1: Wrong
[10]	irst1	Indicates if the previous state's version is a warm boot upgrade version 0: Not a warm boot upgrade version 1: Warm boot upgrade version
[9]	fallback1	Indicates if the previous state was a fallback operation 0: No 1: Yes
[8]	valid1	Last status valid 0: Invalid 1: Valid
[7]	1’b0	Reserved

Bit	Item	Description
[6]	timeout0	Current state timeout 0: Watchdog not timed out 1: Watchdog timeout
[5]	rbcrc_err0	Current state readback CRC error 0: Correct 1: Wrong
[4]	crc_err0	CRC error of the current status 0: Correct 1: Wrong
[3]	id_err0	ID error of the current status 0: Correct 1: Wrong
[2]	irst0	Indicates if the current state's version is a warm boot upgrade version 0: Not a warm boot upgrade version 1: Warm boot upgrade version
[1]	fallback0	Indicates if the current state was a fallback operation 0: No 1: Yes
[0]	valid0	Current status valid 0: Invalid 1: Valid

3.3.2.21 Bitstream 0 Address Register (FADR0R)

Bitstream 0 start address in the Flash, byte addressing.

Table 3-22 Bitstream 0 Address Register Description

Bit	Item	Initial Value	Description
[31:24]	faddr0[31:24]	8’d0	The high 8 bits of the bitstream 0 start address, for 4-byte mode
[23:0]	faddr0[23:0]	24’d0	Bitstream 0 start address

3.3.2.22 Bitstream 1 Address Register (FADR1R)

Bitstream 1 start address in the Flash, byte addressing.

Table 3-23 Bitstream 1 Address Register Description

Bit	Item	Initial Value	Description
[31:24]	faddr1[31:24]	8’d0	The high 8 bits in the bitstream 1 start address, for 4-byte mode
[23:0]	faddr1[23:0]	24’d0	Bitstream 1 start address

3.3.2.23 Bitstream 2 Address Register (FADR2R)

Bitstream 2 start address in the Flash, byte addressing.

Table 3-24 Bitstream 2 Address Register Description

Bit	Item	Initial Value	Description
[31:24]	faddr2[31:24]	8’d0	The high 8 bits in the bitstream 2 start address, for 4-byte mode
[23:0]	faddr2[23:0]	24’d0	Bitstream 2 start address

3.3.2.24 Bitstream 3 Address Register (FADR3R)

Bitstream 3 start address in the Flash, byte addressing.

Table 3-25 Bitstream 3 Address Register Description

Bit	Item	Initial Value	Description
[31:24]	faddr3[31:24]	8’d0	The high 8 bits in the bitstream 3 start address, for 4-byte mode
[23:0]	faddr3[23:0]	24’d0	Bitstream 3 start address

3.3.2.25 Version Fallback Register (FALLBACKR)

The version fallback register specifies the number of retries for the application bitstream and the number of attempts for version fallback.

Table 3-26 Version fallback Register Description

Bit	Item	Initial Value	Description
[31:8]	num_fallback	24’d1	Version fallback attempt count
[7:0]	num_retry	8’d0	Application bitstream retry count

3.4 Bitstream Formats

General bitstream formats are listed in this section.

General bitstream formats are shown in the table below.

Table 3-27 General Bitstream Formats

FFFFFFFF	Padding words (100)
.....	
FFFFFFFF	
000000AA	Bus Bit Width Auto Detection
08100020	
FFFFFFFF	Padding words (10)
.....	
FFFFFFFF	
01332D94	Synchronization
AEC00001	Type 1 packet header: Write RCR control register
xxxxxxxx	Data: Operation code and data of RCR instruction

AB000001	Type 1 packet header: Write to SBPIR
xxxxxxxx	Data: SBPI Master mode configuration control options
A0000000	10 NOP type 1 packet headers
.....	
A0000000	
ABC00001	Type 1 packet header: Write to IRSTCTRLR
xxxxxxxx	Data: Contents of IRSTCTRLR
AC000001	Type 1 packet header: Write to IRSTADRR
xxxxxxxx	Data: Contents of IRSTADRR
A8800001	Type 1 packet header: Write to CMDR
00000000	Data: NOP command
A8800001	Type 1 packet header: Write to CMDR
00000001	Data: RSTCRC command
A8400001	Type 1 packet header: Write to IDR
xxxxxxxx	Data: IDCODE
AC400001	Type 1 packet header: Write to WATCHDOG register
xxxxxxxx	Data: WATCHDOG
ADC00001	Type 1 packet header: Write to CMASKR
xxxxxxxx	Data: Contents of CMASKR
A8C00001	Type 1 packet header: Write to CTRL0R register
xxxxxxxx	Data: Contents of CTRL0R
AE400001	Type 1 packet header: Write to OPTION0R
xxxxxxxx	Data: Contents of OPTION0R
AE800001	Type 1 packet header: Write to OPTION1R
xxxxxxxx	Data: Contents of OPTION1R
AE000001	Type 1 packet header: Write to FALLBACK register
xxxxxxxx	Data: Contents of FALLBACK
A8800001	Type 1 packet header: Write to CMDR
00000002	Data: SWITCH command
A0000000	10 NOP type 1 packet headers
.....	
A0000000	
AAC00001	Type 1 packet header: Write to ADRR
xxxxxxxx	Data: Contents of ADRR
A8800001	Type 1 packet header: Write to CMDR
00000004	Data: WCMEM command
A9400000	Type 1 packet header: Write to CMEMIR
{3'b010, 2'b01, 27'dx}	Type 2 packet header: Write operation
xxxxxxxx	Data: Contents written to configuration memory
.....	

xxxxxxxx	
A0000000	20 NOP type 1 packet headers
.....	
A0000000	
A8000001	Type 1 packet header: Write to CRCR
xxxxxxxx	Data: CRC value
ADC00001	Type 1 packet header: Write to CMASKR
xxxxxxxx	Data: Contents of CMASKR
A9000001	Type 1 packet header: Write to CTRL1R
xxxxxxxx	Data: Contents of CTRL1R
A8800001	Type 1 packet header: Write to CMDR
00000009	Data: GUP command
A8800001	Type 1 packet header: Write to CMDR
00000007	Data: SWAKEUP command
A8000001	Type 1 packet header: Write to CRCR
xxxxxxxx	Data: CRC value
A8800001	Type 1 packet header: Write to CMDR
0000000B	Data: DESYNC command
A0000000	100 NOP type 1 packet headers
.....	
A0000000	

3.5 Download Cable Speed

When using a USB Cable for downloading, including JTAG download and SPI Flash programming, the default speed is 10 MHz.

The download speed can be set in the [Fabric Configuration] by clicking the [Connect To Server] icon.

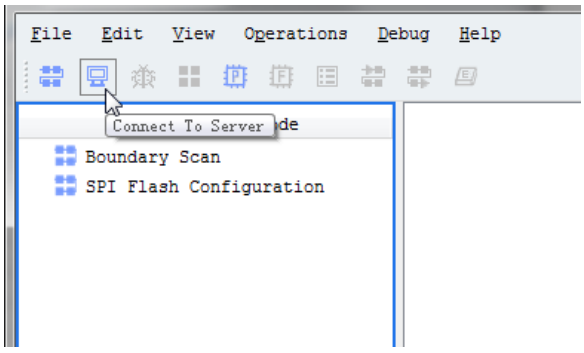

Figure 3-7 Fabric Configuration

If not connected, connect first as shown in the left figure below; if already connected, click [Next] as shown in the right figure below:

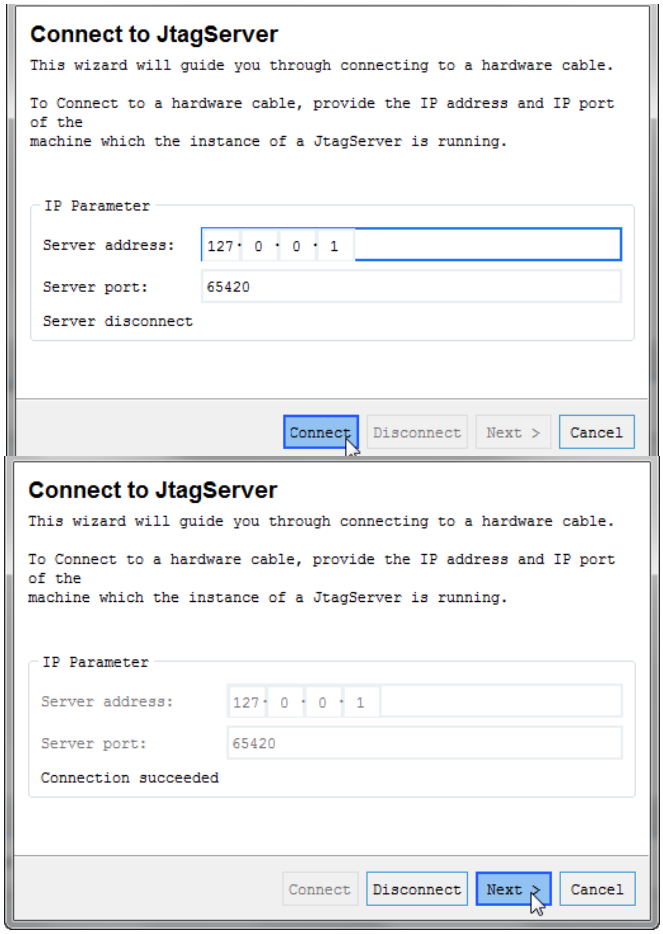

Figure 3-8 Connect To Cable

Users can select the USB Cable frequency in the [TCK Frequency] dropdown menu.

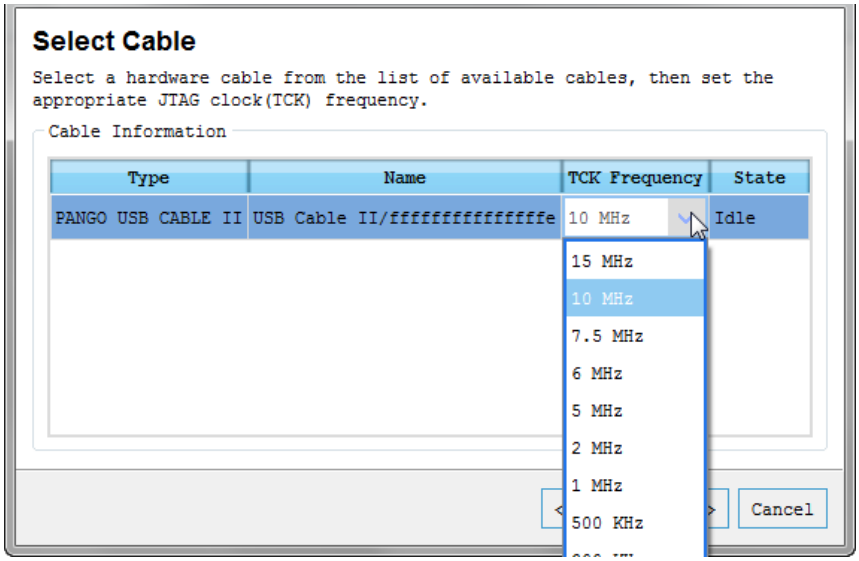

Figure 3-9 USB Cable Speed Settings

3.6 Configuration Speeds

For the maximum download speed in various modes, refer to the "***DS02001_Logos Family***

FPGAs Datasheet".

3.6.1 JTAG Mode

For downloading in the JTAG mode or downloading via a download cable, the maximum configuration frequency is 15 Mhz due to limitation by the cable speed.

3.6.2 Configuration Time

The configuration time required for different configuration modes can be roughly estimated based on the configuration clock frequency, bitstream size, and configuration bit width, as shown in the following formula:

$$\text{Configuration time (s)} \approx \frac{\text{Bitstream size (bit)}}{\text{Clock frequency (Hz)} \times \text{Configuration bit width}}$$

Where the configuration bit width for JTAG mode is 1.

3.7 Multi-function Configuration

In the main mode, the Logos Family FPGAs (PGL12G/22G/22GS) support multi-function configuration.

Users can store up to four independent functional bitstreams to an external Flash. Each function can be any type of bitstream (regular bitstream, compressed bitstream, or encrypted bitstream). The bitstream for which function to configure can be selected by setting the value of pins VS[1:0]. See the table below for details.

Table 3-28 Correspondence between VS[1:0] Values and Bitstreams

VS[1:0] Value	**Bitstream**
2’b00	Function bitstream 0
2’b01	Function bitstream 1
2’b10	Function bitstream 2
2’b11	Function bitstream 3

The multi-function bitstream function jump program enables the VS[1:0] pins to control the start addresses of up to four functions. After jumping to the specified function, disable the VS[1:0] pins.

Table 3-29 Multifunction Jump Program

FFFFFFFF	Padding words (100)	Function jump program
.....		

FFFFFFFF		
000000AA	Bus Bit Width Auto Detection	
08100020		
FFFFFFFF	Padding words (10)	
.....		
FFFFFFFF		
01332D94	Synchronization	
AB000001	Type 1 packet header: Write to SPIR	
00000xxx	Data: SPI master mode operation code	
A0000000	10 NOP type 1 packet headers	
.....		
A0000000		
ABC00001	Type 1 packet header: Write to IRSTCTRL register	
xxxxxxxx	Data: Contents of IRSTCTRL	
AC000001	Type 1 packet header: Write to IRSTADRR	
xxxxxxxx	Data: Contents of IRSTADRR	
ACC00001	Type 1 packet header: Write to FADDR0	
xxxxxxxx	Data: Contents of FADDR0	
AD000001	Type 1 packet header: Write to FADDR1	
xxxxxxxx	Data: Contents of FADDR1	
AD400001	Type 1 packet header: Write to FADDR2	
xxxxxxxx	Data: Contents of FADDR2	
AD800001	Type 1 packet header: Write to FADDR3	
xxxxxxxx	Data: Contents of FADDR3	
A8800001	Type 1 packet header: Write to CMDR	
0000000F	Data: IRST command	
	Bitstream 0	Function 0
	Bitstream 1	Function 1
	Bitstream 2	Function 2
	Bitstream 3	Function 3

The operation flow is as follows:

1. Load the function jump program from Flash0 address.
2. Load the function from the Flash address specified by the bitstream address register selected by VS[1:0].
3. Change the value of VS[1:0] to trigger a hard reset.
4. Repeat steps 1, 2, and 3.

If there is an error during the loading of the application bitstream, load the golden bitstream of the function selected by VS[1:0] from the Flash address specified by the bitstream address register

selected by VS[1:0], or fallback to the previous version of the application bitstream.

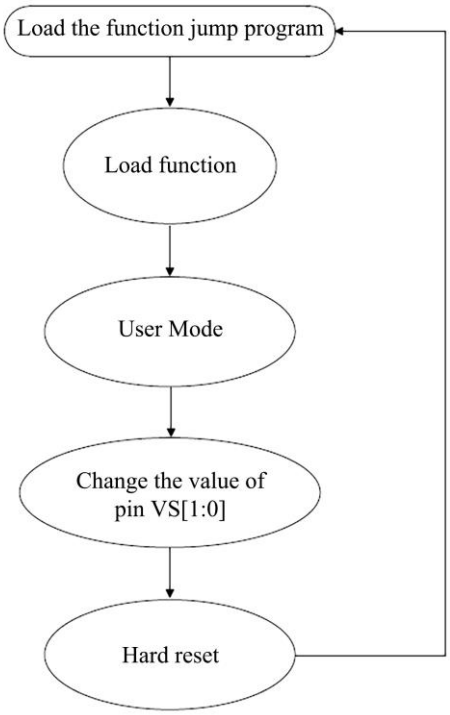

Figure 3-10 Jump Process

3.8 Remote Upgrade

The Logos Family FPGAs support remote upgrades via Master SPI and Master BPI interfaces. During remote upgrades, user logic receives the bitstream from a remote location via communication protocols (such as TCP/IP, PCI, UDP, and UART) or proprietary interfaces. Program the bitstream to internal or external Flash via the user SPI interface.

During remote upgrades, the bitstream can be programmed to internal/external Flash through the JTAG interface; external pins VS[1:0] are generally not used for control, and the enable bit for the VS[1:0] pins in register IRSTCTRLR should be set to 0.

3.8.1 An application bitstream

3.8.1.1 Bitstream

The Logos Family FPGAs identify valid data through a sync word (32’h01332D94), and data can be recognized or masked by adding or removing the sync word before the data. In the remote upgrade bitstream, this serves as a switch to control whether the subsequent jump program is valid.

The bitstream during remote upgrade is shown as follows:

Table 3-30 Remote Upgrade Bitstream

FFFFFFFF	Padding words (1023)	Application bitstream switch program Located in the first sector of the Flash
.....		
FFFFFFFF		
01332D94	Application bitstream switch	
A0000000	NOP type 1 packet header	Application bitstream jump program Located in the second sector of the Flash
AB000001	Type 1 packet header: Write to SPIR	
00000xxx	Data: SPI master mode operation code	
A0000000	10 NOP type 1 packet headers	
.....		
A0000000		
ABC00001	Type 1 packet header: Write to IRSTCTRL register	
xxxxxxxx	Data: Contents of IRSTCTRL	
AC000001	Type 1 packet header: Write to IRSTADRR	
xxxxxxxx	Data: Contents of IRSTADRR	
A8800001	Type 1 packet header: Write to CMDR	
0000000F	Data: IRST command	
A0000000	NOP type 1 packet header (1005)	
.....		
A0000000		
	Golden bitstream	Starting from the third sector of the Flash Occupies integer sectors
	Application bitstream	Starting from the sector following the golden bitstream

3.8.1.2 Operation Flow

Load the application bitstream switch program and the application bitstream jump program from Flash address 0.

Load the application bitstream from the Flash address specified by IRSTCTRLR.

The JTAG or user erases the application bitstream switch program.

The JTAG or user programs Flash to update the application bitstream.

The JTAG or user programs Flash to enable the application bitstream switch.

JTAG warm boot or IPAL warm boot or cold boot or power-up reset or JTAG instruction reset.

If JTAG warm boot or IPAL warm boot, repeat steps 2, 3, 4, 5, 6. If a cold boot occurs (power-up reset/hard reset/JTAG instruction reset), repeat steps 1, 2, 3, 4, 5, 6.

If there is an error during the application bitstream loading process, load the golden bitstream from Flash address 0.

The flowchart is shown below:

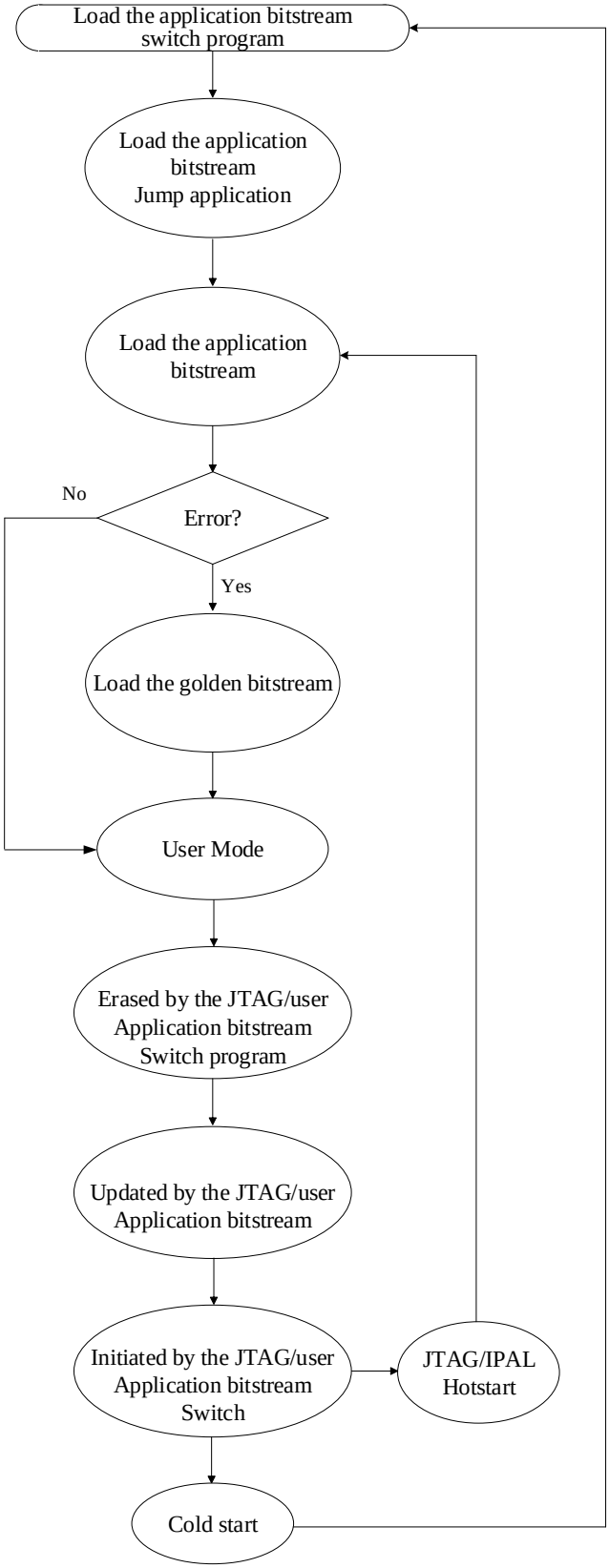

Figure 3-11 Remote Upgrade Process for a Single Application Bitstream

3.8.2 Multiple Application Bitstreams

3.8.2.1 Bitstream

The format for remotely upgrading multiple application bitstreams is shown as follows:

Table 3-31 Format of Multiple Application Bitstreams

FFFFFFFF	Padding words (1023)	Application bitstream 0 switch program Located in the first sector of the Flash
.....		
FFFFFFFF		
01332D94	Application bitstream 0 switch	
A0000000	NOP type 1 packet header	Application bitstream 0 jump program Located in the second sector of the Flash
AB000001	Type 1 packet header: Write to SPIR	
00000xxx	Data: SPI master mode operation code	
A0000000	10 NOP type 1 packet headers	
.....		
A0000000		
ABC00001	Type 1 packet header: Write to IRSTCTRL register	
xxxxxxxx	Data: Contents of IRSTCTRL	
AC000001	Type 1 packet header: Write to IRSTADRR	
xxxxxxxx	Data: Contents of IRSTADRR	
A8800001	Type 1 packet header: Write to CMDR	
0000000F	Data: IRST command	
A0000000	NOP type 1 packet header (1005)	
.....		
A0000000		
FFFFFFFF	Padding words (1023)	Application bitstream 1 switch program Located in the 3rd sector of the Flash
.....		
FFFFFFFF		
01332D94	Application bitstream 1 switch	
A0000000	NOP type 1 packet header	Application bitstream 1 jump program Located in the 4th sector of the Flash
AB000001	Type 1 packet header: Write to SPIR	
00000xxx	Data: SPI master mode operation code	
A0000000	10 NOP type 1 packet headers	
.....		
A0000000		
ABC00001	Type 1 packet header: Write to IRSTCTRL register	
xxxxxxxx	Data: Contents of IRSTCTRL	
AC000001	Type 1 packet header: Write to IRSTADRR	
xxxxxxxx	Data: Contents of IRSTADRR	
A8800001	Type 1 packet header: Write to CMDR	

0000000F	Data: IRST command	
A0000000	NOP type 1 packet header (1005)	
.....		
A0000000		
FFFFFFFF	Padding words (1023)	Application bitstream 2 switch program Located in the 5th sector of the Flash
.....		
FFFFFFFF		
01332D94	Application bitstream 2 switch	
A0000000	NOP type 1 packet header	Application bitstream 2 jump program Located in the 6th sector of the Flash
AB000001	Type 1 packet header: Write to SPIR	
00000xxx	Data: SPI master mode operation code	
A0000000	10 NOP type 1 packet headers	
.....		
A0000000		
ABC00001	Type 1 packet header: Write to IRSTCTRL register	
xxxxxxxx	Data: Contents of IRSTCTRL	
AC000001	Type 1 packet header: Write to IRSTADRR	
xxxxxxxx	Data: Contents of IRSTADRR	
A8800001	Type 1 packet header: Write to CMDR	
0000000F	Data: IRST command	
A0000000	NOP type 1 packet header (1005)	
.....		
A0000000		
	Golden bitstream	Starting from the 7th sector of the Flash Occupies integer sectors
	Application bitstream 0	Starting from the sector following the golden bitstream When using the version fallback function, the old version of the application bitstream must be retained when updating to a new version. That is, the new and old bitstreams cannot use the same sector
	Application bitstream 1	
	Application bitstream 2	

3.8.2.2 Operation Flow

Load the application bitstream switch program and the application bitstream jump program from FLASH address 0.

Load the application bitstream from the FLASH address specified by IRSTCTRLR.

The JTAG or user erases the old version of the application bitstream switch program.

The JTAG or user programs FLASH to update the application bitstream. If the new version of the application bitstream is already in FLASH, there is no need to update.

The JTAG or user programs the FLASH to update the new version of the application bitstream jump program. If the starting address of the new version of the application bitstream remains unchanged, there is no need to update.

JTAG or user programs FLASH to enable the new version of the application bitstream switch.

JTAG warm boot/IPAL warm boot/cold boot (power-up reset/hard reset/JTAG instruction reset).

If JTAG warm boot/IPAL warm boot, repeat steps 2, 3, 4, 5, 6, 7. If a cold boot occurs (power-up reset/hard reset/JTAG instruction reset), repeat steps 1, 2, 3, 4, 5, 6, 7.

If there is an error during the application bitstream loading process, load the golden bitstream from Flash address 0 or fallback to the previous version of the application bitstream. The flowchart is shown below:

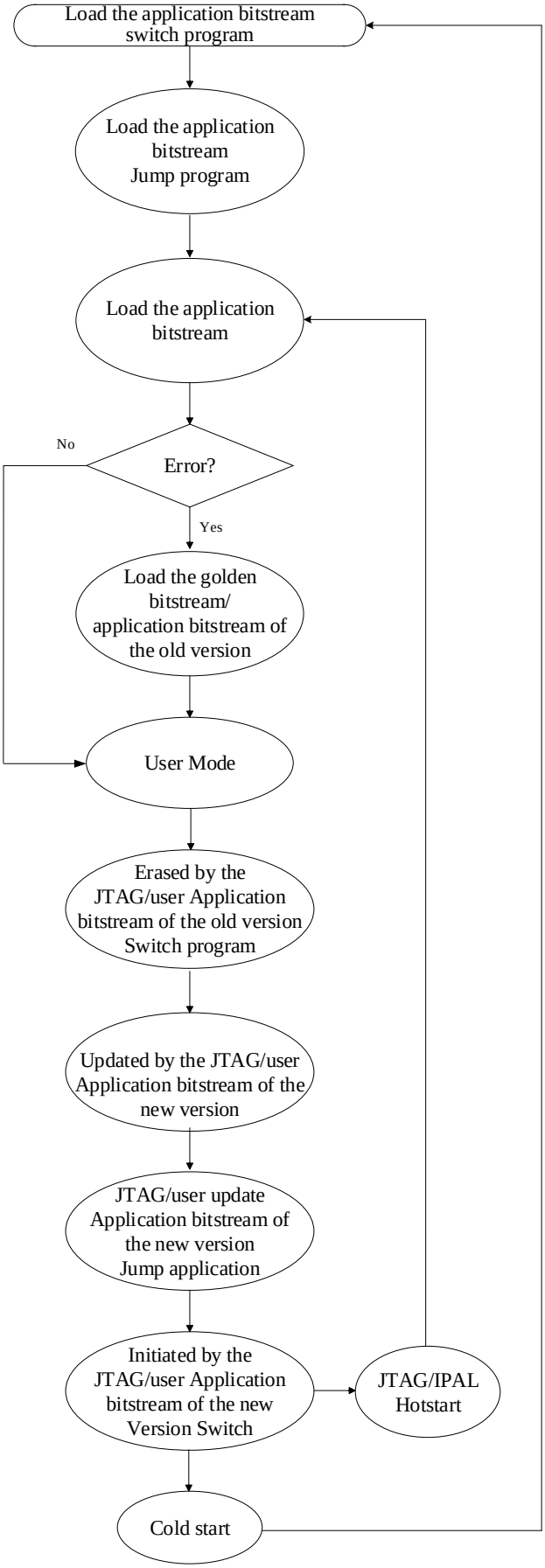

Figure 3-12 Remote Upgrade Process for Multiple Application Bitstreams

3.9 Multi-boot

When using the multi-boot function, the external Flash must support at least two sets of bitstreams. Bitstream 0 is either the golden bitstream version or the application bitstream version, with a starting address of 0, while other bitstreams are application bitstreams. If any errors occur during the configuration process, reset circuits other than the version fallback circuit, and reload the golden bitstream or the previous version of the application bitstream. If the golden bitstream or the previous version's application bitstream also contains errors, then it will not reset. Instead, while setting INIT_FLAG_N to 0, FCS_N is set to 1, ending the SPI master mode operation.

When using the multi-boot function, it is necessary to enable the watchdog timer.

During configuration, the following errors will trigger a version fallback.

1. Incorrect device ID
2. CRC error
3. Watchdog timeout

3.9.1 Golden bitstream initialization system

1. Load the golden bitstream from Flash address 0.
2. Perform Flash operations via JTAG or by the user. If the application bitstream for multi-boot is already in Flash, Flash operations are not necessary.
3. JTAG warm boot.
4. Load the application bitstream from the Flash address specified by register IRSTCTRLR.
5. If there is an error while loading the application bitstream, load the golden bitstream from Flash address 0 or fallback to the previous version's application bitstream.

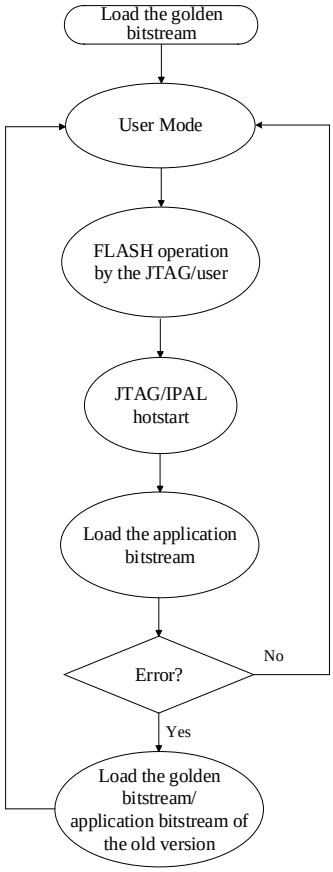

Figure 3-13 Golden Bitstream Loading Process

3.9.2 Application bitstream initialization system

1. Load the golden bitstream from Flash address 0.
2. The golden bitstream's embedded warm boot loads the application bitstream from the Flash address specified by register IRSTCTRLR.
3. If there is an error while loading the application bitstream, load the golden bitstream from Flash address 0 or fallback to the previous version's application bitstream.

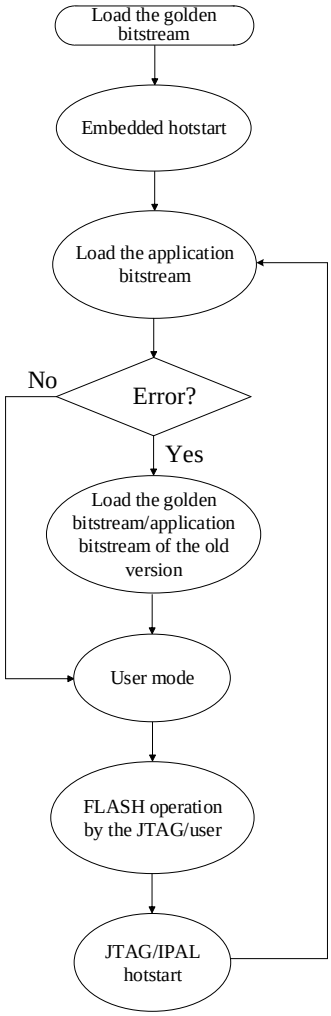

Figure 3-14 Application bitstream loading process

3.9.3 Without golden bitstream

If the Flash supports a maximum of two bitstreams, when the version fallback function is enabled, both bitstreams are application bitstreams.

1. Load the application bitstream from Flash address 0.
2. Perform Flash operations via JTAG or by the user. If the application bitstream for multi-boot is already in Flash, Flash operations are not necessary.
3. JTAG warm boot or IPAL warm boot.
4. Load the application bitstream from the Flash address specified by register IRSTCTRLR.
5. If there is an error while loading the application bitstream, fallback to the previous version's application bitstream.

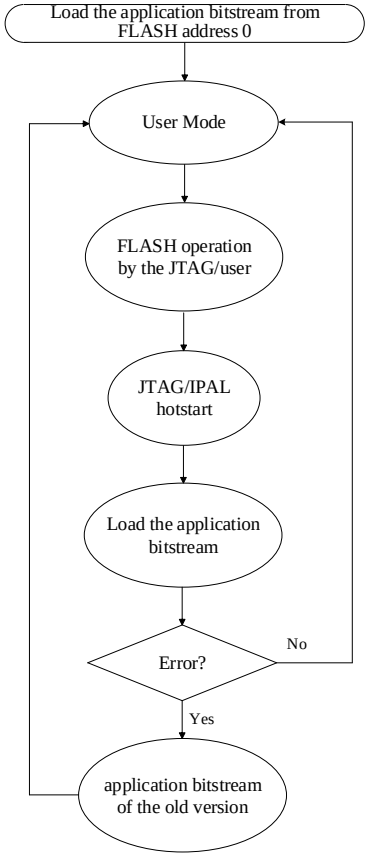

Figure 3-15 Loading process without golden bitstream

3.9.4 Warm boot

Before sending instructions to the device, synchronization is required to ensure the device can correctly interpret the instructions. If sending instructions via the Slave Parallel interface, a bus width detection instruction must also be sent. In a multi-boot application, the golden bitstream includes a instruction to write to IRSTADRR, with the address being the starting address of the application bitstream, eliminating the need to include IRSTADRR register operations during a warm boot. If the multi-boot does not include a golden bitstream, the user needs to write the starting address of the bitstream to be loaded during warm boot into the IRSTADRR register as part of the warm boot process. If a warm boot instruction is embedded within the golden bitstream, it is not possible to upgrade to the golden bitstream via warm boot within the application bitstream initialization system.

The complete warm boot process is shown in the following table.

Table 3-32 Warm boot process

FFFFFFFF	Padding words (100)
.....	
FFFFFFFF	

000000AA	Bus Bit Width Auto Detection
08100020	
FFFFFFFF	Padding words (10)
.....	
FFFFFFFF	
01332D94	Synchronization
AC000001	Type 1 packet header: Write to IRSTADRR (optional)
xxxxxxxx	Data: Contents of IRSTADRR (optional)
A8800001	Type 1 packet header: Write to CMDR
0000000F	Data: IRST command
A8800001	Type 1 packet header: Write to CMDR
0000000B	Data: DESYNC command
A0000000	100 NOP type 1 packet headers
.....	
A0000000	

3.9.5 Watchdog

The watchdog timer is used to detect timeouts during configuration and in user mode.

The watchdog timer counts down by 1 for every 512 system clock cycles (software default set to 100MHz). When the watchdog timer count reaches 0, a watchdog timeout flag is generated.

During the primary mode configuration process, a watchdog timeout triggers a version fallback.

The watchdog timer is disabled during and after the version fallback process. After the version fallback is successfully completed, a warm boot can lift the prohibition on the watchdog.

Chapter 4 Readback Operation

Logos FPGA supports reading back both standard and compressed bitstreams through JTAG and Slave Parallel interfaces.

4.1 JTAG Interface Readback

Before reading back through the JTAG interface, it is necessary to detect the wakeup_over signal in the IR or the status register to ensure that the device is awake before proceeding with the readback. Typically, readback occurs a significant amount of time after the wakeup process has completed.

4.2 Slave Parallel Interface Readback

To perform readback using the Slave Parallel interface, ensure that the interface is enabled or preserved.

If using this interface for readback in configuration mode, ensure that the active configuration mode is set to Slave Parallel mode, i.e., the MODE pin is set to Slave Parallel mode.

If using this interface for readback in user mode, ensure that the active bitstream has the configuration interface preserved (set in the PDS under "Project Setting->Generate Bitstream->Readback->Persist Slave Parallel Pins", not preserved by default). Additionally, the data width used for readback must match the width used during configuration; that is, if configured with Slave Parallel x8 and the configuration interface is preserved, readback must be performed with Slave Parallel x8 and not with x16/x32 data widths.

When performing readback via the Slave Parallel interface, it is necessary to send some instructions to the device according to the timing requirements of the Slave Parallel interface, as described in the readback configuration memory process and readback configuration register process below, where the instruction stream is presented in 32-bit form and must be sent according to
Bit Orders under Different Interface Bit Widths. Once the device correctly receives the instructions, it will output the data on the rising edge of the clock once the data is ready, and the user can then sample the data. The process for readback of configuration registers and configuration memory differs, as described below.

4.2.1 Readback Configuration Memory (with busy port)

Before readback operations via the Slave Parallel interface, it is necessary to detect the wakeup_over signal in the status register to ensure that the device is awake before proceeding with readback. Typically, readback occurs a significant amount of time after the wakeup process has completed.

The readback process with a busy port is as follows:

Table 4-1 Readback Process with Busy Port

Flow	**Instruction Flow (Hex)/Operation**
100 padding bytes	FFFFFFFF ..... FFFFFFFF
Bus Bit Width Auto Detection	000000AA 08100020
10 padding bytes	FFFFFFFF ..... FFFFFFFF
SYNC WORD	01332D94
Type 1 packet header: No operation	A0000000
Write a RCMEM command to the CMDR register	A8800001 00000006
Write the ADRR register	AAC00001 00000000(1)
Write the header for a Type 1 packet to read the CMEMOR register	B1C00000
Write the header for a Type 2 packet for reading data	5xxxxxxx(2)
Switching between read and write (CS_N is 1, clock CFG_CLK continues)	Pull CS_N high, and change RWSEL to read when CS_N is 1
Read the CMEMOR register	Pull CS_N low and sample data
Switching between read and write (CS_N is 1, clock CFG_CLK continues)	Pull CS_N high, and change RWSEL to write when CS_N is 1
Write a DESYNC instruction to CMDR register	A8800001 0000000B
100 NOP Type 1 packet headers	A0000000 ..... A0000000

Notes:

1. Write the readback start address to the ADRR register.
2. The number of 32-bit words for readback.

4.2.2 Readback Configuration Memory (without busy port)

Before readback operations via the Slave Parallel interface, it is necessary to detect the wakeup_over signal in the status register to ensure that the device is awake before proceeding with readback. Typically, readback occurs a significant amount of time after the wakeup process has completed.

When readback is performed without using a busy port, it is necessary to determine whether the data has been fully read back and whether a complete frame has been read back. If the data has been completely read back, proceed to the next step to synchronize operations. If the data has not been completely read back and the current frame is not fully read back, continue the readback process. If the data has not been fully read back but the current frame has, the Slave Parallel interface will set CS_N to 1 and wait for at least 20 configuration clock cycles. The PGL22G device does not support this operation process.

Table 4-2 Readback Process without Busy Port

Flow	**Instruction Flow (Hex)/Operation**
100 padding bytes	FFFFFFFF ..... FFFFFFFF
Bus Bit Width Auto Detection	000000AA 08100020
10 padding bytes	FFFFFFFF ..... FFFFFFFF
SYNC WORD	01332D94
Type 1 packet header: No operation	A0000000
Write a RCMEM command to the CMDR register	A8800001 00000006
Write the ADRR register	AAC00001 00000000(1)
Write the header for a Type 1 packet to read the CMEMOR register	B1C00000
Write the header for a Type 2 packet for reading data	5xxxxxxx(2)
Switch between read and write, and pause operations for at least 200 configuration clock cycles (CS_N is 1, and the clock CFG_CLK continues)	Pull CS_N high, and change RWSEL to read when CS_N is 1
Read CMEMOR register (during readback, determine whether the readback is complete and whether it is a frame boundary)	Pull CS_N low and sample data
Switching between read and write (CS_N is 1, clock CFG_CLK continues)	Pull CS_N high, and change RWSEL to write when CS_N is 1
Write a DESYNC instruction to CMDR register	A8800001 0000000B
100 NOP Type 1 packet headers	A0000000 ..... A0000000

Notes:

1. Write the readback start address to the ADRR register.

2. The number of 32-bit words for readback.

4.2.3 Readback Configuration Register (with busy port)

For readback operations with a busy port, the process for reading back the configuration register is shown as follows:

Table 4-3 Readback Configuration Register Process with busy port

Flow	**Instruction Flow (Hex)/Operation**
100 padding bytes	FFFFFFFF ..... FFFFFFFF
Bus Bit Width Auto Detection	000000AA 08100020
10 padding bytes	FFFFFFFF ..... FFFFFFFF
Synchronization	01332D94
Type 1 packet header: No operation	A0000000
Write NOP command to CMDR register	A8800001 A0000000
Write the header of a type 1 packet to read a specified register	Bxxxxxxx(1)
Switching between read and write (CS_N is 1, clock CFG_CLK continues)	Pull CS_N high and change RWSEL to read when CS_N is 1 (2)
Read the specified register	Pull CS_N low and sample data
Switching between read and write (CS_N is 1, clock CFG_CLK continues)	Pull CS_N high and change RWSEL to write when CS_N is 1 (2)
Write a DESYNC instruction to CMDR register	A8800001 0000000B
100 NOP Type 1 packet headers	A0000000 ..... A0000000

Notes:

1. According to Packet Types, fill in the

Configuration register address that needs to be read back, as well as the number of 32-bit words to be read back.

4.2.4 Readback Configuration Register (without busy port)

For readback operations without a busy port, the process for reading back the configuration register is as follows:

Table 4-4 Readback Configuration Register Process without Busy Port

Flow	**Instruction Flow (Hex)/Operation**
100 padding bytes	FFFFFFFF ..... FFFFFFFF
Bus Bit Width Auto Detection	000000AA 08100020
10 padding bytes	FFFFFFFF ..... FFFFFFFF
Synchronization	01332D94
Type 1 packet header: No operation	A0000000
Write NOP command to CMDR register	A8800001 A0000000
Write the header of a type 1 packet to read a specified register	Bxxxxxxx(1)
Switch between read and write, and wait for at least 20 configuration clock cycles (CS_N is 1, and the clock CFG_CLK continues)	Pull CS_N high and change RWSEL to read when CS_N is 1
Read the specified register	Pull CS_N low and sample data
Switching between read and write (CS_N is 1, clock CFG_CLK continues)	Pull CS_N high and change RWSEL to write when CS_N is 1
Write a DESYNC instruction to CMDR register	A8800001 0000000B
100 NOP Type 1 packet headers	A0000000 ..... A0000000

Notes:

1. According toPacket Types, fill in theConfiguration register address that needs to be read back, as well as the number of 32-bit words to be read back.

Chapter 5 Readback CRC

Readback CRC uses a 32-bit CRC algorithm and is performed during readback. During data readback, a 32-bit data undergoes one CRC calculation.

After reading back the configuration memory twice, examine the readback CRC results.

If the readback CRC is incorrect, the user may choose the next step. Such as reconfiguration, etc.

If the readback CRC is correct, the configuration memory can be read back repeatedly to continue the readback CRC. After each readback of the configuration memory, examine the readback CRC results.

The readback CRC function can be implemented using IP.

Chapter 6 SEU Detection

The FPGA device supports SEU 1-bit correction and 2-bit detection. The ECC algorithm uses the SECDED algorithm, grouped by frame.

SEU detection and correction can be performed through the internal parallel interface.

Read back one frame of data from the configuration memory.

Examine the SEU results.

If there are no errors, continue to read back the next frame.

If a single bit error occurs, dynamically reconfigure the current frame, then continue to read back the next frame.

If a double bit error occurs, the user should choose the next step of operation. For example, the user can dynamically reconfigure the current frame or choose to reconfigure and so on.

SEU detection functionality can be implemented using IP cores.

Chapter 7 Design Protection

7.1 eFuse

eFuse is a one-time programmable non-volatile memory used to store device configuration information, UID, and other user-programmable bits. Other user-programmable bits include a 256-bit AES-CBC encryption key and a 32-bit user fuse.

The value of an unprogrammed fuse is 0, and the value after programming is 1. All user-programmable bits can be configured through software. See the included document "***Fabric_Configuration_User_Guide***" for specific operations.

When configuring eFuse, the VCCEFUSE voltage should be powered up according to the recommended power-up sequence, and after configuration, the voltage can be maintained or grounded after powering down according to the power-down sequence requirements; it should not be left floating.

The 64-bit UID is uniformly programmed at the time of device manufacture and can be read via the JTAG interface or UID interface, see UID Interface. The 32-bit user fuse can be read through the JTAG interface or the User Fuse interface, see User eFuse Bit Interface.

7.2 Bitstream Encryption

Malicious cloning of user designs by competitors is one of the common issues during the use of FPGAs. To address this issue, the Logos Family PGL12G/22G/22GS/50G/50H/100H devices incorporate a bitstream encryption function to prevent malicious cloning of user designs. Please note that when configuring the encrypted bitstream through the Slave Parallel configuration interface, it only supports a data width of 8; other configuration modes have no usage restrictions. The PGL25G does not support bitstream encryption protection.

7.2.1 Scenarios without bitstream encryption protection

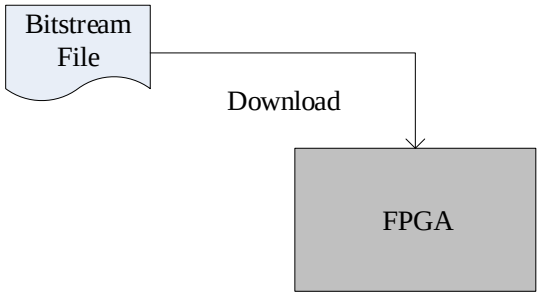

Figure 7-1 Scenario without bitstream encryption

As shown in the figure above, the Logos Family FPGAs are SRAM-based FPGAs, which requires loading the bitstream file from an external source every time it is powered on. The user's bitstream file is typically stored in an external storage element of the FPGA, such as Flash memory.

Malicious cloners can obtain the bitstream file in various ways. After obtaining the bitstream file, malicious cloners can directly load it onto their own purchased identical FPGA, thereby replicating the exact same FPGA functionality as the user.

7.2.2 Scenarios with bitstream encryption protection

To effectively protect users' intellectual property, the Logos Family FPGAs offer bitstream encryption capabilities.

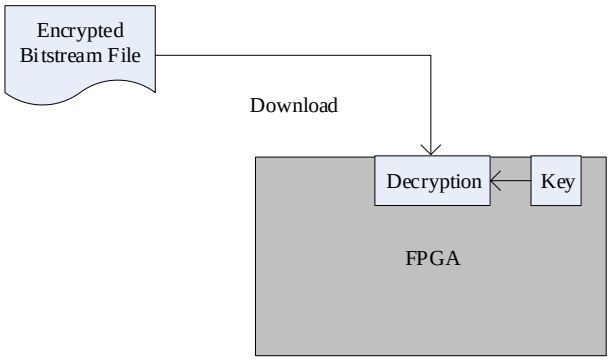

Figure 7-2 Bitstream encryption scenario

As shown in the figure above, the bitstream file itself is encrypted using the AES-256 encryption algorithm. When the encrypted bitstream file is loaded onto the Logos Family FPGAs, it must first be decrypted with the key previously stored inside the chip. Here, the key of the encrypted bitstream file must match the key stored inside the Logos FPGA; otherwise, the Logos FPGA cannot enter its normal working state.

The key inside the device is pre-written by the user via JTAG, and can be written to locations such as eFuse or a temporary key register used for testing. eFuse is one-time programmable (OTP) and

non-volatile, meaning that once the key information is programmed, it will be permanently retained. Furthermore, the device's key functions a read protection function; once read protection is enabled, the key that is written cannot be read back. When writing the key into eFuse, the download tool will enable key read protection by default. Programming eFuse requires following a specific power sequence. For details, see Section 2.5 of the "***DS02001_Logos Family FPGAs Datasheet***".

With the use of bitstream encryption protection, users are no longer troubled by malicious copying. Even if a malicious copier obtains the bitstream file (encrypted), they do not know the key, and thus cannot load the bitstream file into a device that does not contain the corresponding key.

7.2.3 Bitstream Encryption Protection Usage Process

The main usage process for bitstream encryption protection includes:

1. Selecting a key, with attention to the user needing to properly safeguard the key;
2. When generating bitstream files with the user's PDS, use the selected key to produce an encrypted bitstream file;
3. The user utilizes the PDS's built-in Fabric configuration to burn the key into the Logos Family products. Afterward, users can confidently use the encrypted bitstream file to load the Logos FPGA that has undergone the process described in 3).

The steps for generating an encrypted bitstream file (sbit) are as follows: for example, using PDS 2021.1.sp7.2, right-click on the [Generate Bitstream] option in PDS and select the [Configure] option to set it up, as shown in the figure below. It can also be selected through the [Process] in the menu.

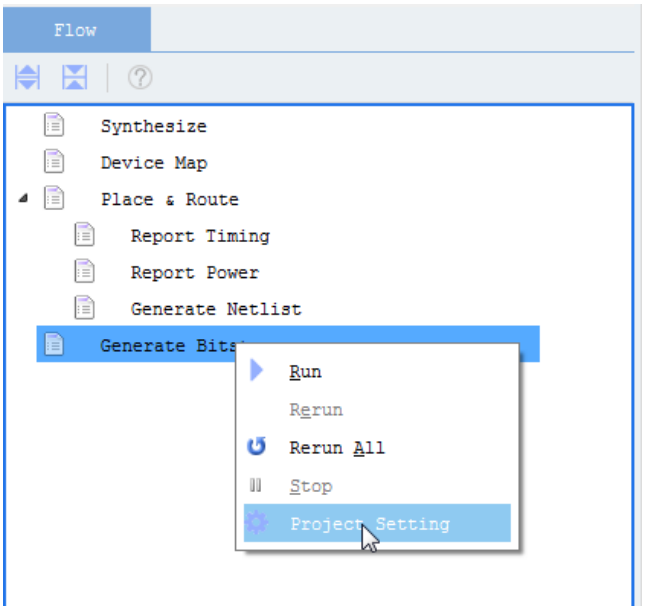

Figure 7-3 Configure Interface

The pop-up window is shown in the figure below:

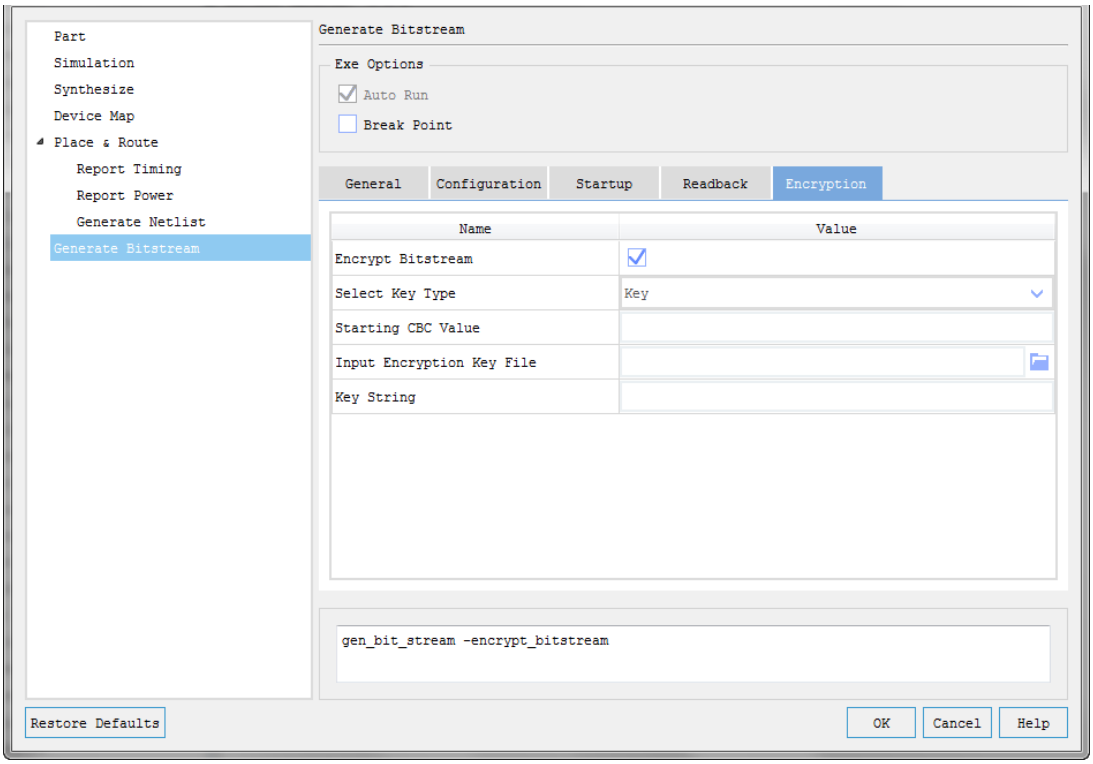

Figure 7-4 Configure Settings Option

Among these, the [Encrypt Bitstream] option determines whether to encrypt the bitstream file; with [Create Bit File] under [General] configured to Yes, selecting [Encrypt Bitstream] will generate an encrypted sbit file and nky file (containing the initial CBC value of 128 bits and a 256-bit key required for encryption). Users can edit the CBC and key strings or select an nky file; if the user does not specify, an nky file will be randomly generated.

[Select Key Type] configures whether to use an internal KEY, i.e., programmed into eFuse, with the default value being Key. When selecting Temporary Key, the corresponding nky file must be downloaded before each bitstream download; selecting Key, this operation only needs to be performed once, and then the bitstream can be directly downloaded without the key being lost after the device is powered down;

[Starting CBC Value] The user manually enters the initial CBC string;

[Input Encryption Key File] Select an nky file; if a key file is chosen here, the software will encrypt using the key file, regardless of whether the user has entered a key string in the options;

[Key String] The user manually enters the key string;

After the settings are complete, click [OK] to exit the configuration window. Run Generate Bitstream again to produce the encrypted bitstream file.

7.3 UID

A significant risk in the reconfigurable device industry is the design piracy and illegal overproduction. Therefore, to prevent such illegal activities, the UID was introduced. Each device has a unique identifier corresponding to it, which is uniquely determined at the time of the device's manufacture. Users can read through UID Interface (instantiating GTP_UDID) and the JTAG interface, and incorporate the results obtained after processing with their unique encryption algorithm into the programming bitstream. After each bitstream load, the device enters user mode, where the user logic reads the UID and processes it with a unique encryption algorithm to compare with the results in the previous programming bitstream; if there is a difference, then the device will not function properly.

The 64-bit UID is stored in eFuse and is programmed uniformly when the device is manufactured. Each time the device is powered on, the UID from the eFuse is automatically read into the register for user access at any time.

Chapter 8 User Logic Interfaces

8.1 Internal Slave Parallel Interface

GTP_IPAL_E1 provides an internal slave parallel interface, as well as Readback CRC and SEU interfaces. Users can operate the internal slave parallel interface by instantiating this GTP, and can also conveniently perform Readback CRC and SEU operations using the Readback CRC and SEU interfaces. This section mainly introduces the internal slave parallel interface. For the use of Readback CRC and SEU, please refer to the document "***AN020006_Logos Family FPGAs SEU Application Guide***".

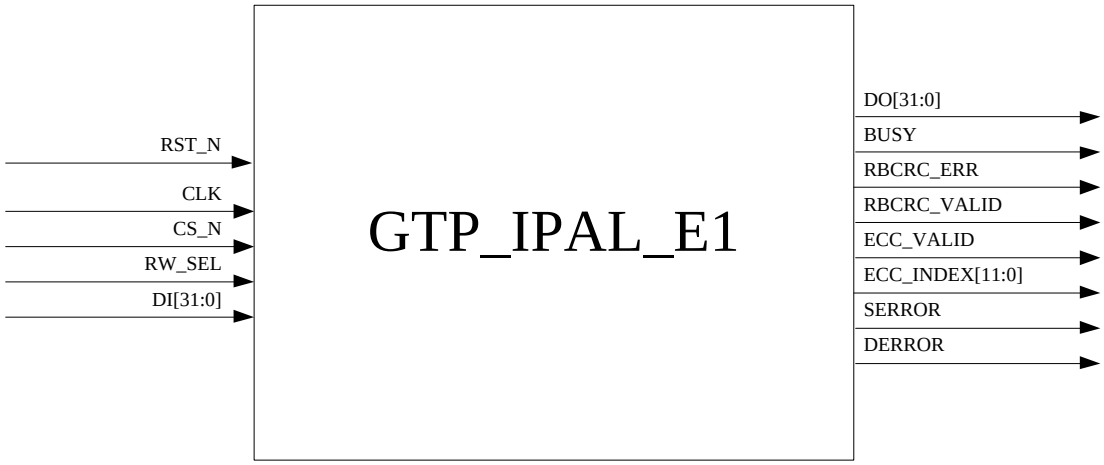

Figure 8-1 GTP_IPAL_E1 Port Diagram

8.1.1 Port List

Table 8-1 GTP_IPAL_E1 Port List

Port	I/O	Bit width	Description
CLK	I	1	Internal slave parallel interface clock
RST_N	I	1	Reset signal, active-low, used only for simulation (1)
Internal slave parallel interface			
CS_N	I	1	Chip select signal, active-low
RW_SEL	I	1	Read/write selection: 0 for write, 1 for read;
DI	I	32	Data input
DO	O	32	Data output
BUSY	O	1	Readback busy flag, 1 indicates data is not ready

Port	I/O	Bit width	Description
Readback CRC Interface			
RBCRC_ERR	O	1	Readback CRC error flag, active-high.
RBCRC_VALID	O	1	Readback CRC valid flag, active-high, lasts for one clock cycle.
SEU Detection Interface			
ECC_VALID	O	1	ECC valid flag, active-high.
ECC_INDEX	O	12	Single-bit error address index
SERROR	O	1	SEU detection single-bit error flag
DERROR	O	1	SEU detection double-bit error flag

Note: 1. This signal is only used to reset the interface to its initial state during simulation; there is no such port on the actual hardware, and operating this port is neither necessary nor will it have any effect.

8.1.2 Parameter Definitions

Table 8-2 GTP_IPAL_E1 Parameter List

Parameter Name	Parameter Type	Valid Values	Function Description
IDCODE	Binary	0~32'hffffffff	Device IDCODE, used only for simulation
DATA_WIDTH	String	"X8", "X16", "X32"(1)	Parallel bit width
SIM_DEVICE	String	"PGL12G", "PGL22G", "PGL25G", "PGL50G", "PGL50H", "PGL100H"	Device selection, used only for simulation

Note:

1. PGL25G, 50G, 50H, 100H only support X8, X16 bit widths, and do not support X32 bit width.

8.1.3 Interface timing

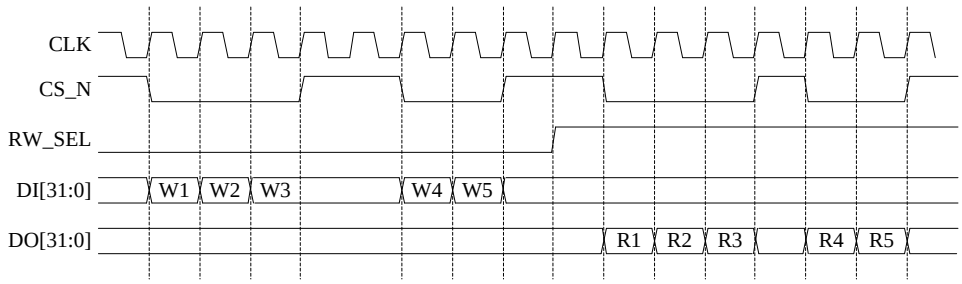

Figure 8-2 GTP_IPAL_E1 Internal Slave Parallel Interface Read/Write Interface Timing

Note:

1. The RW_SEL signal transition can only occur when CS_N is high.

8.1.4 Instantiation template

```
GTP_IPAL_E1 #(
    .DATA_WIDTH("X8"),
    .IDCODE('b10101010101010100101010101010101),
    .MEM_DEPTH(1616)
) <InstanceName> (
    .DO(),
    .ECC_INDEX(),
    .DI(),
    .BUSY(),
    .DERROR(),
    .ECC_VALID(),
    .RBCRC_ERR(),
    .RBCRC_VALID(),
    .SERROR(),
    .CLK(),
    .CS_N(),
    .RST_N(),
    .RW_SEL()
);
```

8.2 UID Interface

8.2.1 Port List

Table 8-3 GTP_UDID Port List

Port	I/O	Bit width	Function Description
DI	I	1	Serial data input
DO	O	1	Serial data output
SE	I	1	Enables data shift
LOAD	I	1	Data registers parallel load UID CODE
CLK	I	1	Clock

8.2.2 Parameter List

Table 8-4 GTP_UDID Parameter List

Parameter Name	Parameter Type	Valid Values	Function Description
UDID_WIDTH	Integer	64	UID length, for simulation only (1)
UDID_CODE	Binary	0~96'hffffffffffffffffffffffff	Chip identity code, for simulation only

Note:

1. This interface model applies to Pango Compa, Logos, and Logos2 family devices. The UID length varies between families, with a default value of 64. Therefore, to maintain consistency between simulation and board operation, the default value should be kept 64 for Logos devices.

8.2.3 Interface timing

The UID is 64-bit long.

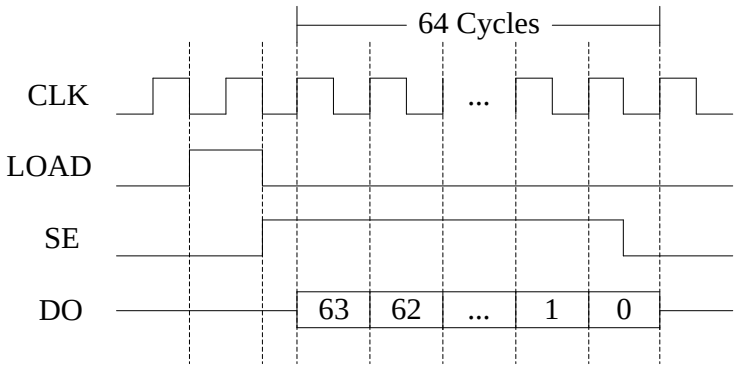

Figure 8-3 Read UID Timing

Users can extend the bit width of the UID.

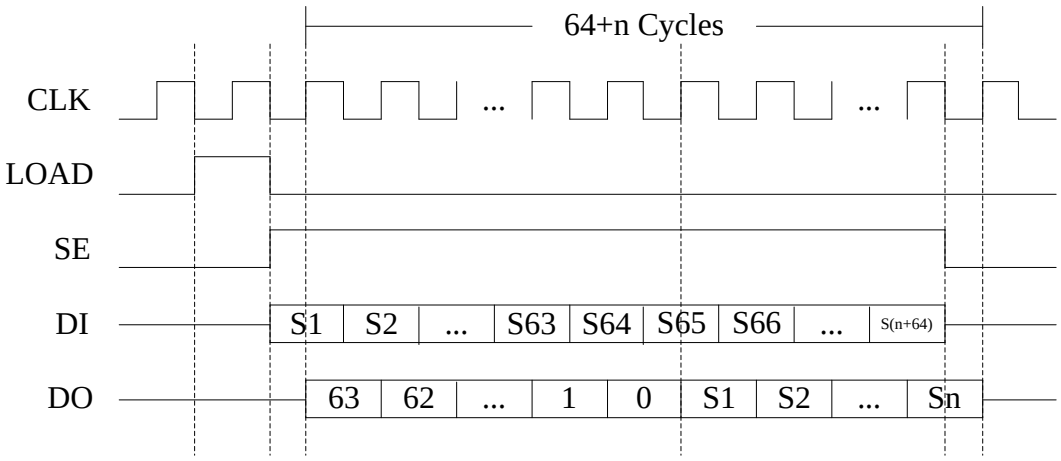

Figure 8-4 Extended UID Bit Width

8.2.4 Instantiation template

The Verilog instantiation template for GTP_UDID is shown as follows.

```
GTP_UDID #(
    .UDID_CODE('b00000000000000000000000000000000000000000000000000000000000000000)
) <InstanceName> (
    .DO(),
    .CLK(),
    .DI(),
    .LOAD(),
    .SE()
);
```

8.3 User eFuse Bit Interface

User logic can read the eFuse user bits by instantiating the eFuse user bit interface (GTP_EFUSECODE), which outputs the value of the 32-bit user bits in an asynchronous parallel manner.

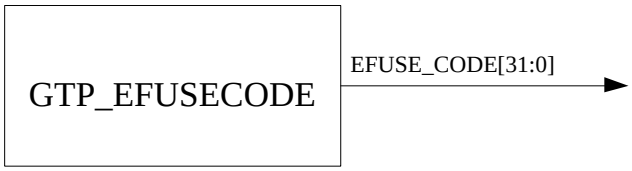

Figure 8-5 GTP_EFUSECODE Port Diagram

8.3.1 Port List

Table 8-5 GTP_EFUSECODE Port List

Port	I/O	Bit width	Function Description
EFUSE_CODE	O	32	Efusecode output data bus

8.3.2 Parameter List

Table 8-6 GTP_EFUSECODE Parameter List

Parameter Name	Parameter Type	Valid Values	Function Description
SIM_EFUSE_VALUE	Binary	32’h0~32’hFFFF_FFFF	EFUSE_CODE simulation value

8.3.3 Instantiation template

The Verilog instantiation template for GTP_EFUSECODE is shown as follows.

```
GTP_EFUSECODE #(
    .SIM_EFUSE_VALUE('b00010010001101000101011001111000)
) <InstanceName> (
    .EFUSE_CODE()
);
```

Disclaimer

Copyright Notice

Disclaimer

1. This document only provides information in stages and may be updated at any time based on the actual situation of the products without further notice. The Company assumes no legal responsibility for any direct or indirect losses caused by improper use of this document.
2. This document is provided "as is" without any warranties, including but not limited to warranties of merchantability, fitness for a particular purpose, non-infringement, or any other warranties mentioned in proposals, specifications, or samples. This document does not grant any explicit or implied intellectual property usage license, whether by estoppel or otherwise.
3. The Company reserves the right to modify any documents related to its family products at any time without prior notice.