Logos2 Family FPGA Remote Upgrade Application Guide

(AN04002, V1.0)

(26.11.2020)

Shenzhen Pango Microsystems Co., Ltd.

Revisions History

Document Revisions

Version	**Date of Release**	**Revisions**
V1.0	26.11.2020	Initial release.

About this Manual

Terms and Abbreviations

Abbreviations	Meaning
CPLD	Complex Programmable Logic Device
CRAM	Configuration Random Access Memory
JTAG	Joint Test Action Group
CCS	Configuration Control System
TAP	Test Access Port
PDS	Pango Design Suite

Table of Contents

Tables

Figures

Chapter 1 Overview

1.1 Introduction

This document serves as the application manual for the remote upgrade scheme of the Logos2 Family FPGA devices, provided by Shenzhen Pango Microsystems Co., Ltd. This document primarily introduces the main functions, interface definitions, interface timing, supported devices, and reference designs for the remote upgrade of the Logos2 Family FPGA devices.

1.2 Main Functions

The remote upgrade of Logos2 Family FPGA devices is essentially to complete code version upgrades or fallbacks of the chip remotely without affecting the current working state of the chip. Its functions include utilizing the Serial Port to "upgrade the flash application bitstream, read back the applied bitstream from the flash chip, verify the correctness of the applied bitstream written to the flash, and hot-start the FPGA" while keeping the IO port state unchanged.

Note: The IO port state remains unchanged during the remote upgrade of the chip.

1.3 Design Information

Table 1-1 Remote Upgrade Product Information

Remote Upgrade Product Information	
Supported Devices	PGL100H
Supported User Interface	Serial Port
Provided Design Files	
Remote Upgrade Reference Designs	Verilog files fdc file
Development Tools	
Design Tools	Pango Design Suite 2020.2-SP2

Note: The upgrade bitstream is a simple LED flashing light in the example.

Chapter 2 Function Description

The Logos Family FPGA supports 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 external Flash via the user SPI interface.

The Logos2 Family FPGA devices 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. Users can use JTAG to write the merged bitstream (switch program + jump program + golden bitstream + applied bitstream) into the external flash chip and remotely update the applied bitstream.The Logos2 Family FPGA devices support 1 to 3 applied bitstreams. After the chip powers up, the Configuration Control System (CCS) loads the corresponding bitstream file by controlling the status (on or off) of the corresponding switch program. During the update of the applied bitstream, if an abnormal situation such as power-down occurs, the CCS, when re-powered up, will automatically load the golden bitstream to ensure that the FPGA has a version of bitstream to use and can start normally.

The diagram for remotely upgrading the Logos2 Family FPGA device application is shown in Figure 2-1.

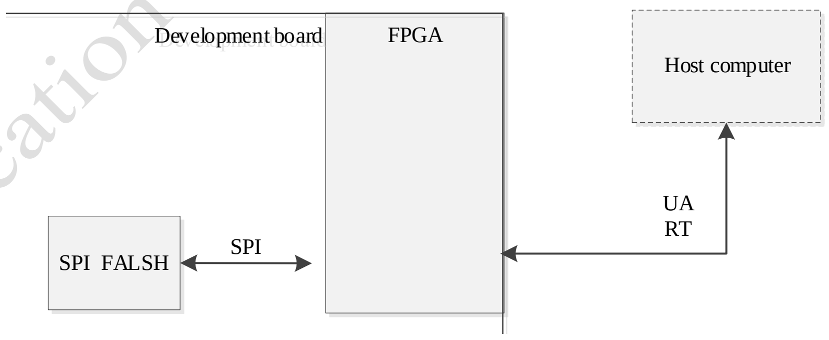

Figure 2-1 Diagram for Remotely Upgrading Logos2 Family FPGA Device Application

2.1 Module Introduction

The block diagram of remotely upgrading the Logos2 Family FPGA device system is shown in Figure 2-2. The main functions of the modules in the red dashed box on the right include communication with the host computer, data caching, and command parsing. The red solid box on the left contains the common modules that mainly control read/write flash and hotreset. If the communication interface in the remote upgrade scheme is replaced with Ethernet or other interfaces, the modules in the solid box on the left can be universal[1], while the modules in the dashed box on the right need to be adapted to the communication protocol.

The host computer can read and write the remote upgrade-related configuration registers and status registers through the serial port, issuing various control commands. The host computer sends the bitstream file while the serial port module (uart_top.v) converts the serial protocol data and sends it to the subsequent data control module (data_ctrl.v) for caching. The spi flash control module (spi_top.v) reads the bitstream data and write it into the flash chip. After writing the data, it sends a completion indicator to the IPAL control module (ipal_ctrl.v). The IPAL control module writes the hotreset command into the GTP_IPAL_E2 module for hotreset. After restarting, the Configuration Control System (CCS) loads the latest applied bitstream to complete the remote upgrade.

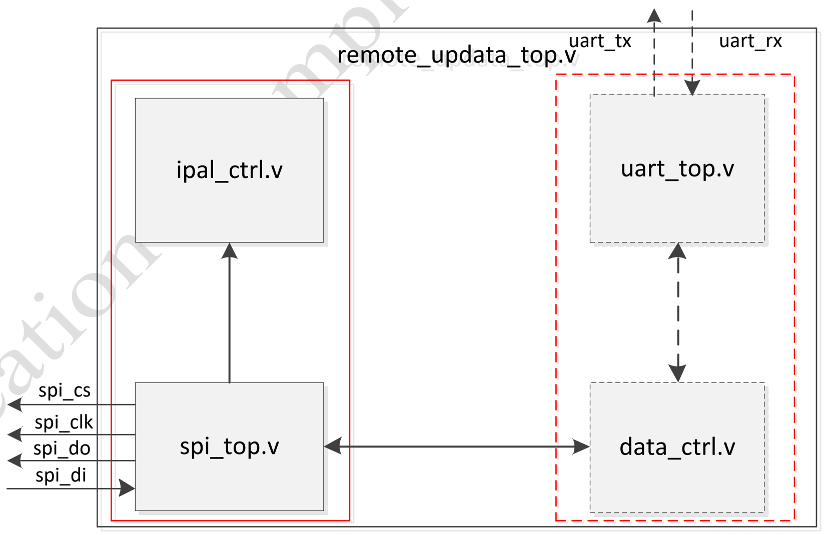

Figure 2-2 Block Diagram of System Remote Upgrade

1 The process of remote upgrade and the data interaction method between the common module and the preceding module cannot be changed.

2.1.1 Serial Port Module (uart_top.v)

The serial port module is mainly responsible for serial communication with the host computer. The default baud rate is 115200bps and cannot be modified dynamically; To modify it, alter the corresponding parameters during instantiation. The currently supported baud rates are: 2400bps, 4800bps, 9600bps, 19200bps, 38400bps, and 115200bps. The block diagram of serial port module is shown in Figure 2-3 below.

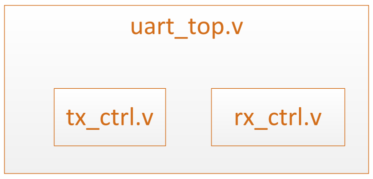

Figure 2-3 Block Diagram of Serial Port Module

2.1.2 Data Control Module (data_ctrl.v)

The data control module has two main functions: parsing the commands sent by the host computer and controlling the read/write registers; caching bitstream data and aligning it to a 4KB boundary for convenience in subsequent processing.

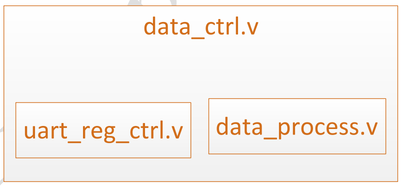

Figure 2-4 Block Diagram of Data Control Module

2.1.3 spi flash Control Module (spi_top.v)

The spi flash control module completes spi bus control and writes bitstream data to an external flash chip. The spi_driver.v module is responsible for executing single instructions (reading configuration registers, erasing sectors, programming pages, etc.). The spi_top.v module decomposes the read/write bitstream commands into multiple single instructions and writes them into FIFO. Then the spi_driver.v module executes the instructions sequentially.

Notice: The spi flash chip used in this case is Micron N25Q256. For specific instructions, refer to the N25Q256 datasheet. If the instructions used by the spi flash chip are incompatible with those of this chip, modifications will be needed.

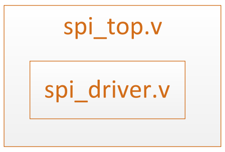

Figure 2-5 Block Diagram of spi flash Control Module

2.1.4 IPAL Control Module (ipal_ctrl.v)

The IPAL control module mainly performs hotreset for Logos2 Family FPGA devices. After the bitstream data is written, the IPAL control module will receive an indicator. If hotreset_en is 1, the module will perform hotreset immediately; if hotreset_en is 0, the module will not perform hotreset immediately. Users can manually write 1 to this register through the host computer and then perform hotreset. The description of the hotreset_en register can be found in the subsequent register description section.

2.2 Description of Common Module Interface Signals

2.2.1 Interface Signal List

Table 2-1 Common Module Interface Signals

Signal Name	Direction	Bit width	Description
Clock and Reset Signals			
sys_clk	input	1	System clock
sys_rst_n	input	1	System reset signal, active-low
External Pin Signals			
spi_cs	output	1	Chip selection pin of spi flash chip
spi_clk	output	1	Clock pin of spi flash chip
spi_dq1	input	1	Data input pin of spi flash chip
spi_dq0	output	1	Data/instruction output pin of spi flash chip
Enable Control Signal			
flash_wr_en	input	1	Write bitstream data enable
flash_rd_en	input	1	Read bitstream data enable
hotreset_en	input	1	Enable hotreset
open_sw_code_en	input	1	Write switch program enable
bs_crc32_ok	input	2	[1]:valid [0]:1'b0,crc32 verify OK; 1'b1,crc32 verify error
Feedback Indicator Signal			
bs_readback_crc	output	32	Read bitstream verification CRC result
bs_readback_crc_valid	output	1	Read bitstream verification CRC result valid

Signal Name	Direction	Bit width	Description
clear_done_ind	output	1	Erase complete indication
bitstream_wr_done	output	1	Write bitstream file complete
bitstream_rd_done	output	1	Read Bitstream file complete
open_sw_code_done	output	1	Write switch program complete
Readback Data Interface			
flash_rd_data	output	8	Read bitstream data
flash_rd_valid	output	1	Read bitstream data valid
flash_rd_data_fifo_afull	input	1	Read bitstream data caching FIFO almost full
Write flash data interface (front-end data caching needs packet FIFO)			
bitstream_fifo_rd_req	output	1	FIFO (used to cache bitstream files written to flash) read request
bitstream_data	input	8	FIFO (used to cache bitstream files written to flash) read out data
bitstream_valid	input	1	FIFO (used to cache bitstream files written to flash) read out data valid
bitstream_eop	input	1	FIFO (used to cache bitstream files written to flash) data packet end marker Each data packet is 256 bytes (1 page) for easier subsequent processing
bitstream_fifo_rd_rdy	input	1	FIFO (used to cache bitstream files written to flash) non-empty

The interface signals between the common module and the front-end module mainly include enable control signal, feedback indicator signal, and read/write data interaction signal. If other interfaces are used to communicate with the host computer, the front-end module needs to adapt to the corresponding communication protocol. The front-end module needs to control the enable signal based on the process and feedback indicator signal. The write data interface requires a packet FIFO with a length of 256 bytes to store data, and the readback data interface uses a regular FIFO to cache data.

2.2.2 Interface Timing Diagram

The control signal and indicator signal are primarily related to the remote upgrade process, while the readback and write data interfaces mainly involve data interaction with the FIFO caches of the front-end module. The timing of the data interaction and that of the control process are relatively independent, so they are explained separately.

The timing diagram for control signal and feedback indicator signal is shown in Figure 2-6. The update process of the bitstream file is described in Section 3.2.2. The timing of the interfaces is easier to understand in combination with the process description in the following text.

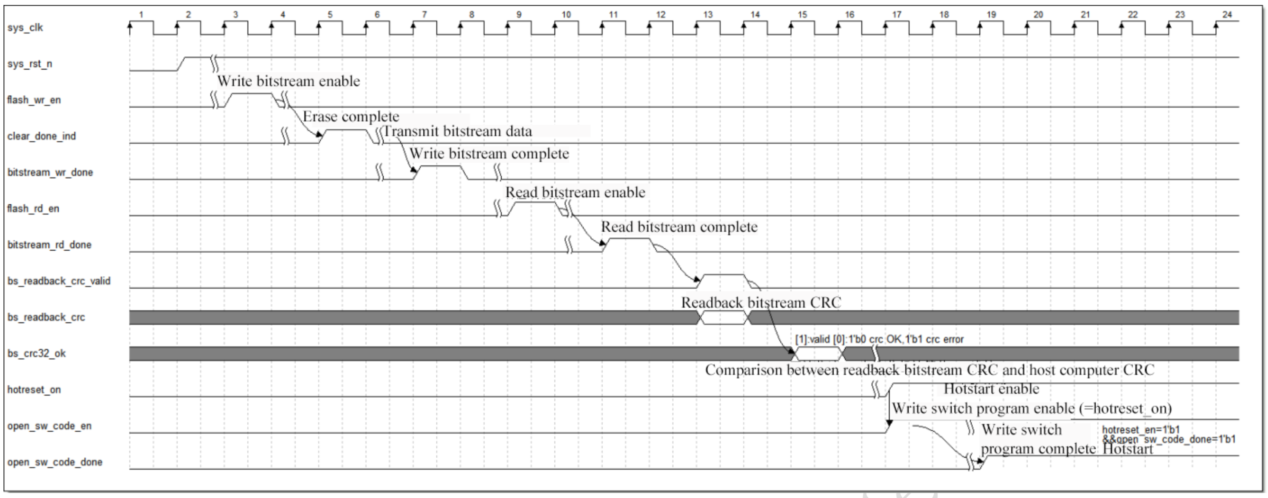

Figure 2-6 Timing Diagram of Control and Feedback Indicator Signal

The timing diagram for the write bitstream data interface is shown in Figure 2-7.

Note: The front-end FIFO needs to meet two conditions:

1. The front-end FIFO must be a packet FIFO with a length of 256 bytes;

2. The read request signal must be aligned with valid data, i.e., when the read request (bitstream_fifo_rd_req) is issued, data is output immediately without delay.

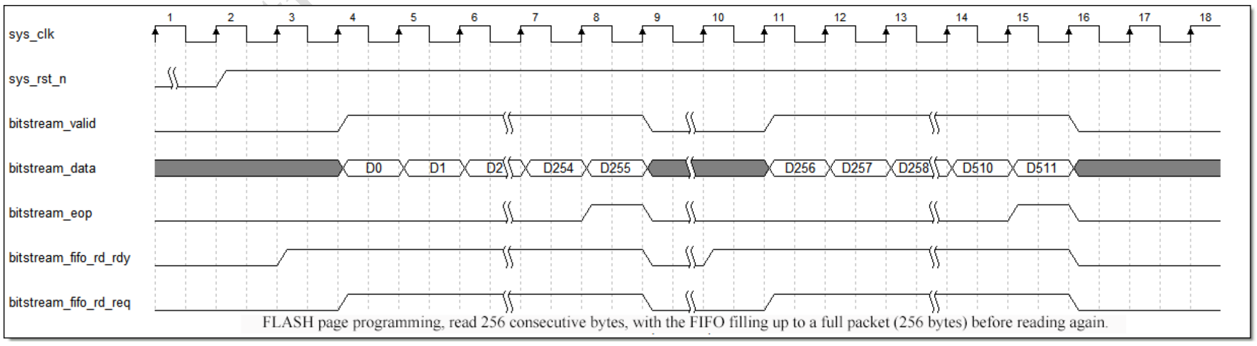

Figure 2-7 Timing Diagram of Write Bitstream Data

The timing diagram of readback bitstream data is shown in Figure 2-8. Read bitstream data from FLASH based on the backpressure control of the front-end module. To facilitate control and maintain efficiency, read one page (256 bytes) of data each time. The read out data will be continuously fed regardless of whether the front-end FIFO is under backpressure or not. If there is backpressure after reading all the data, the read will pause. Therefore, the front-end FIFO should reserve at least 256 bytes of space before it becomes full to commence backpressure.

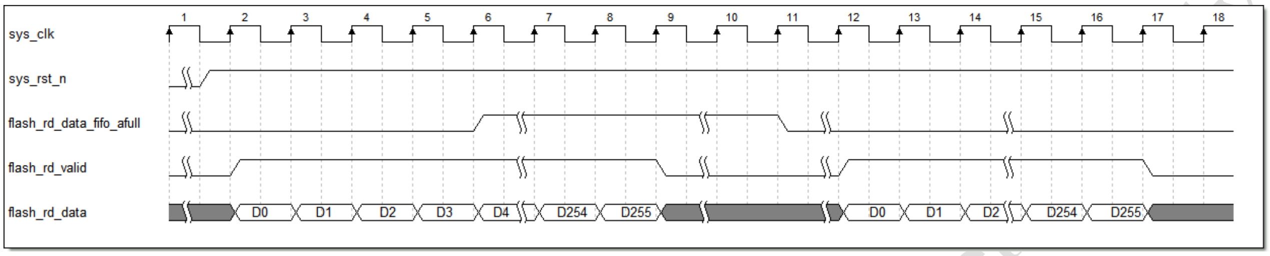

Figure 2-8 Timing of Data Readback

Chapter 3 Reference Designs

3.1 Register Description[2]

This reference design uses the development board PG2L100RD03_A0 of Pango Microsystems as the main control device and UartAssist (the serial port assistant tool on the PC) as the host computer to complete the remote upgrade debugging. As the communication between the host computer and the development board is done via the serial port, there is no distinction between register addresses and data, thereby necessitating a custom set of command codes. The codes are shown in the table below:

Table 3-1 Command Codes

Command Code Issued by Host Computer	**Command Code Uploaded to Host Computer**	**Command Code for Bitstream Write Complete**	**Command Code for Read Address**	**Command Code for Write Address**
32'he7e7_e7e7	8’h55	32'h7e7e_7e7e	{1’b1,addr[6:0]}	{1’b0,addr[6:0]}

3.1.1 Read/Write Operation

Read/write operations are commands issued by the host computer. During both the read and write, the host computer issues the command code of "32'he7e7_e7e7+ address". The address command code is one byte, with the lower 7 bits as the address. The highest bit distinguishes read and write, with 1'b1 indicating read and 1'b0 indicating write. Example: Read register 1, 32'he7e7_e7e7+8'h81; Write register 1, 32'he7e7_e7e7+8'h01+data to be written (1 byte).

When a read register command is issued, the FPGA will return the corresponding register value. For data of erase complete, write bitstream file complete, and CRC complete, a read operation is not required. The FPGA will upload the data in the same format. The return data ranges from 1 to several bytes, depending on the register. The write data bitstream command is introduced in the following section of update process.

2 In this case, the host computer and FPGA communicates with each other via the serial port, and the register configuration and command format are customised to fit the serial port. If other methods are used, adjustments are required.

3.1.2 Register Address Allocation

There are mainly 5 registers and 2 commands, as shown in the table below.

Table 3-2 Register Address Allocation

Item	**Address**	**Read /Write**	**Function Description**
fpga_version	7’h0	R	Version information, 6 bytes, can be customized. Example: 48'h2020_0101_1230.
crc32_cfg	7’h1	W/R	The host computer calculates the CRC of the bitstream file and then configures it into the register; 4 bytes.
test_reg	7’h2	W/R	Test register
crc32_error_ind	7’h3	R	Flag indicating that the CRC result of the read bitstream is not equal to crc32_cfg. 1’b1: Not equal
hotreset_en	7’h4	W/R	hotreset_en [0]: Hotreset enable, active high. If valid, hotreset is initiated after the bitstream update is completed.
wr_bs_status	7’h5	R	wr_bs_status[0]: Erase complete flag, active high; wr_bs_status[4]: Write bitstream file flag, active high.
wr_user_bs_en	7’h11	W	Update applied bitstream file command.
rd_user_bs_en	7’h51	W	Read applied bitstream file command.

The version information register fpga_version and the CRC register crc32_cfg have 6 bytes and 4 bytes respectively. When reading these two registers, all bytes will be returned at once. When writing crc32_cfg, all bytes will also be configured at once (e7 e7 e7 e7 +01+ xx xx xx xx). Other registers only have one byte, so the read/write operations involve only one byte of content.

The return data format is "8'h55+address+return data". Example: The data received when reading version information is 55 00 20 20 01 01 12 30, where 55 is the upload command code, 00 is the address, and 20 20 01 01 12 30 is the value of the version information register.

3.2 Application Description

3.2.1 Storage Format of Bitstream Files

The storage format of the bitstream file in the flash chip is shown in the following figure:

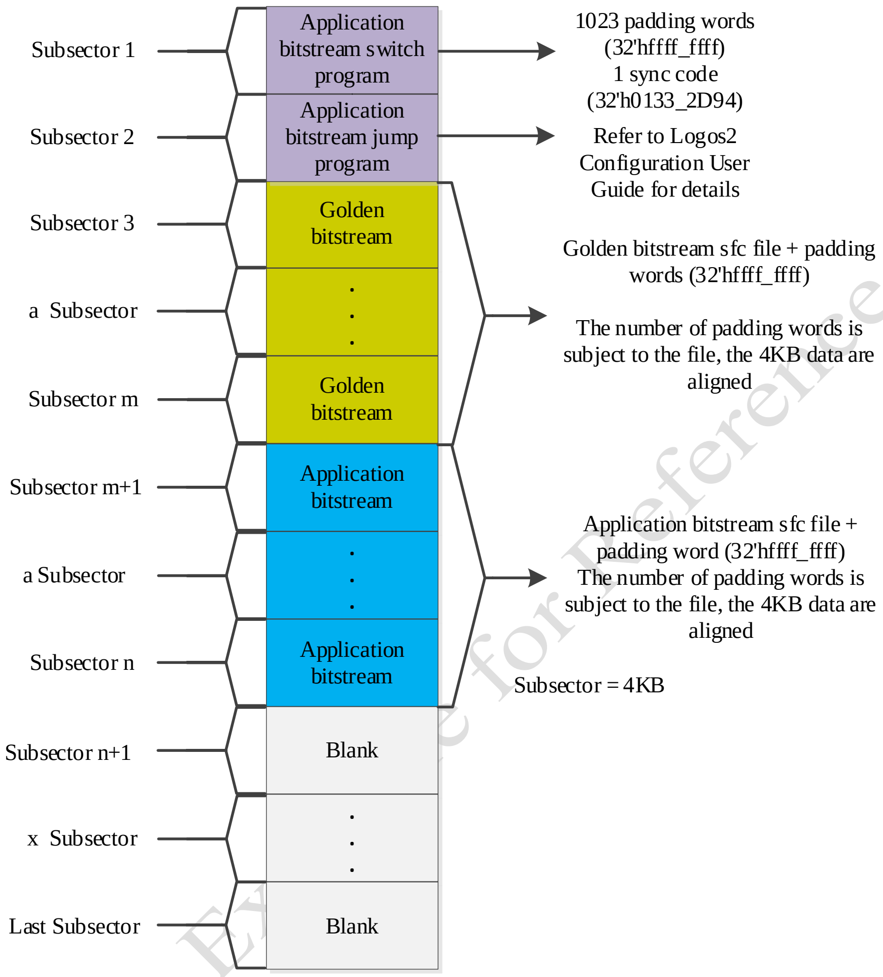
Subsector 1
Subsector 2
Subsector 3
a Subsector
Subsector m
Subsector m+1
a Subsector
Subsector n
Subsector n+1
x Subsector
Last Subsector
Application bitstream switch program
Application bitstream jump program
Golden bitstream
Golden bitstream
Application bitstream
Application bitstream
Blank
Blank
1023 padding words (32'hffff_ffff) 1 sync code (32'h0133_2D94)
Refer to Logos2 Configuration User Guide for details
Golden bitstream sfc file + padding words (32'hffff_ffff)
The number of padding words is subject to the file, the 4KB data are aligned
Application bitstream sfc file + padding word (32'hffff_ffff) The number of padding words is subject to the file, the 4KB data are aligned
Subsector = 4KB

Figure 3-1 Storage Format of Bitstream Files in Flash

Merged bitstream files in flash are divided into 4KB (one subsector) segments. The first 4KB contains the switch programme, consisting of 1023 instances of 32'hffff_ffff and 1 synchronisation code (32'h0133_2d94). The second 4KB is the applied bitstream jump program; The detailed content can be found in the "UG040005 Logos2 Family FPGA Configuration User Guide". The third 4KB begins with the golden bitstream data; any portion not reaching 4KB is filled with 32'hffff_ffff. The next subsector is the applied bitstream data; any portion not reaching 4KB is filled with 32'hffff_ffff. The positions of the golden bitstream and applied bitstream can be set when generating the merged bitstream file, but the golden bitstream must come before the applied bitstream.

3.2.2 Update Process of Bitstream Files

The process of updating bitstream during remote upgrade is shown in the above figure. The update process requires cooperation between the host computer and the FPGA. The host computer sends commands to enable bitstream write, bitstream read, and hotreset, while the FPGA completes the other steps.

The detailed steps are as follows:

1. The host computer sends a write bitstream enable command and waits for the FPGA to erase the switch program and applied bitstream. Upon completion of the erasure, the FPGA sends a completion flag (55 05 01) to the host computer.
2. Once the host computer receives the completion flag for erasure, it sends the end flag of the bitstream file and bitstream (7e 7e 7e 7e). If the host computer receives (55 05 10), it indicates that the applied bitstream write is completed, and the next step can be carried out. Users can read the bitstream for verification, or repeat the first two steps to write the bitstream file again.
3. The host computer sends a read bitstream enable command and reads the bitstream for verification. Upon verification completion, the FPGA reports the verification results (55 03 01/00). (55 03 01) indicates that the verification result is incorrect, and (55 03 00) indicates that the verification result is correct.
4. The host computer sends a command to enable hotreset and loads the new applied bitstream. If the preceding verification result is incorrect, the golden bitstream will be loaded.

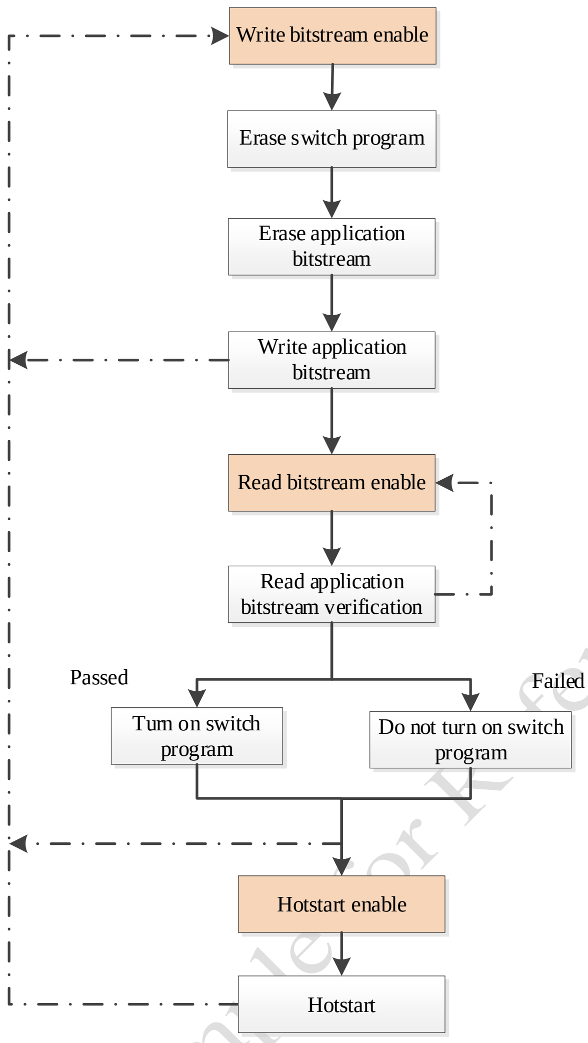
Write bitstream enable
Erase switch program
Erase application bitstream
Write application bitstream
Read bitstream enable
Read application bitstream verification
Passed
Failed
Turn on switch program
Do not turn on switch program
Hotstart enable
Hotstart

Figure 3-2 Update Process of Applied Bitstream

3.2.3 Considerations

1. Versions before 2020.2.SP2 have desynchronization codes, while versions after 2020.2.SP2 do not have desynchronization codes. The current codes are processed without desynchronization codes. Please use 2020.2.SP2 or later versions for generating the merged bitstream.
2. The time required to erase the applied bitstream varies depending on the flash chip. The erasure time for the Micron M25Q256 is approximately 20 seconds, and the bitstream data can only be sent after the erasure is completed.
3. The current application is developed and debugged on the PG2L100RD03_A0 development board, with an SPI CLK frequency of 25 Mhz and without any issues in read/write operations. If read/write errors occur in actual applications, it is necessary to first confirm whether the SPI CLK frequency is too high.

4. In the remote upgrade process, each step must be completed before proceeding to the next step. Otherwise, errors will occur, requiring a reset or power-up to recover.
5. If the serial port assistant mentioned in this case is used, sfc files can not be read directly. If this serial port assistant is used as the host computer, scripts or other tools are needed to extract and send the bitstream data.

Disclaimer