Compact Family CPLD Configuration User Guide

(UG030004, V1.7)

(2023.09.29)

Revisions History

Document Revisions

Version	Date of Release	Revisions
V1.7	2023/09/29	Initial release.

About this Manual

Terms and Abbreviations

Terms and Abbreviations	Meaning
CPLD	Complex Programmable Logic Device
JTAG	Joint Test Action Group
TAP	Test Access Port
I^2C	Inter-Integrated Circuit
APB	Advanced Peripheral Bus
CCS	Configuration Control System
SPI	Serial Peripheral Interface
UID	Unique ID
ISC	In-System Configuration

Table of Contents

Tables

Figures

Chapter 1 Overview

The Compact family CPLD is an SRAM-based programmable logic device with an embedded Flash in the chip for autonomous bitstream loading from within the chip. As the configuration information in the SRAM of the CPLD device will be lost upon power-down, it is necessary to reconfigure the CPLD each time it is powered up.

Configuration is the process of writing the user-designed bitstream into the CPLD, and the entire process is completed by the Configuration Control System (CCS) with the structure as shown in the diagram Figure 1-1, where the user logic connects to seven slave devices through the APB bus. The bitstream can be actively retrieved by the chip from an external SPI Flash or embedded Flash, or downloaded into the chip by an external processor/controller or other master devices.

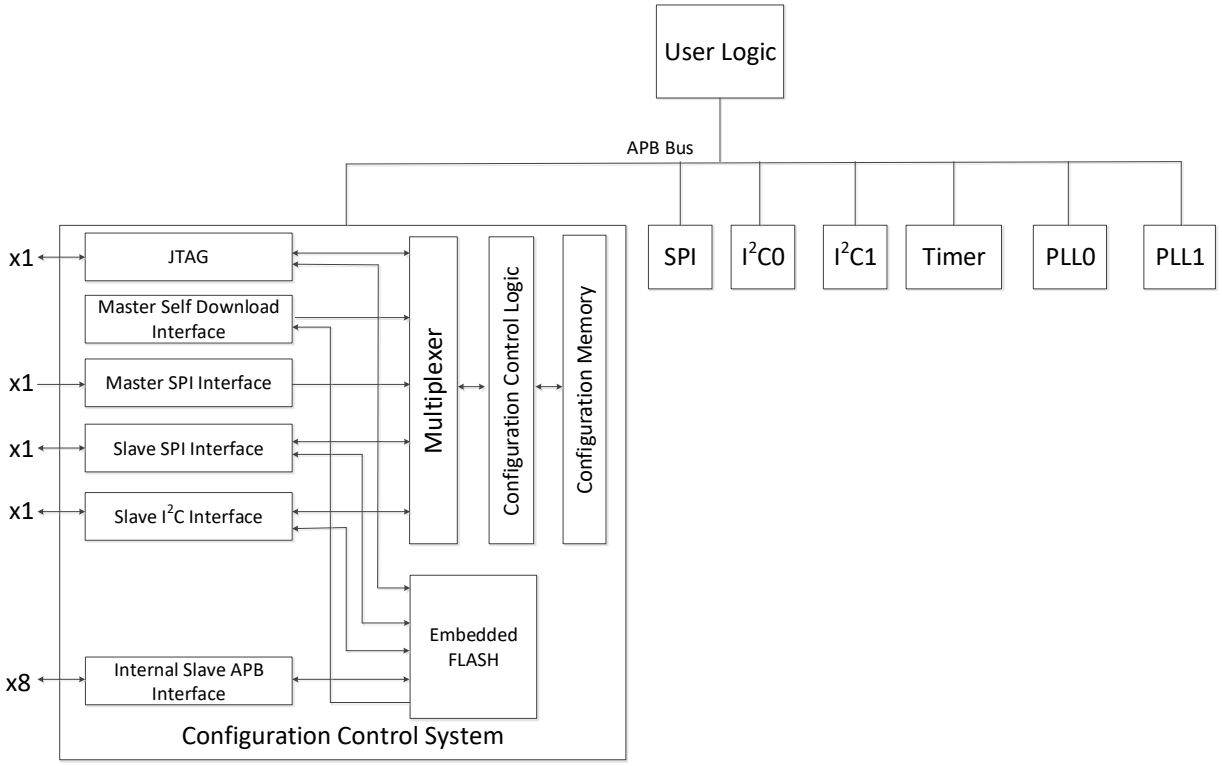

Figure 1-1 Configuration System Structure Diagram

CPLD devices have 5 configuration modes, as detailed below:

- JTAG mode, with data width X1, compliant with IEEE 1149.1 and IEEE 1532 standards; support configuring and reconfiguring as well as reading back normal/compressed bitstreams.
- Master Self Download mode, supports configuring normal and compressed bitstreams.
- Master SPI mode, with data width X1, supports configuring normal and compressed bitstreams.

- Slave SPI mode, with data width X1, supports configuring and reconfiguring both normal and compressed bitstreams.
- Slave I^2C mode, with data width X1, supports configuring and reconfiguring normal bitstreams and compressed bitstreams
- Furthermore, CPLD devices also provide the following features:
- Support online debugging and boundary scan testing
- Watchdog, supporting configuration time-out detection
- Support dual-boot and bitstream version fallback for all 5 configuration modes
- User logic interfaces, including JTAG user instruction interface (SCANCHAIN), UID interface, power controller, user wake-up interface (START), internal slave APB interface, and readback CRC
- Embedded hard cores, including 1 SPI, 2 I^2C, and 1 timer

Chapter 2 Configuration Process

The configuration download procedures for CPLD devices are shown in the following figure:

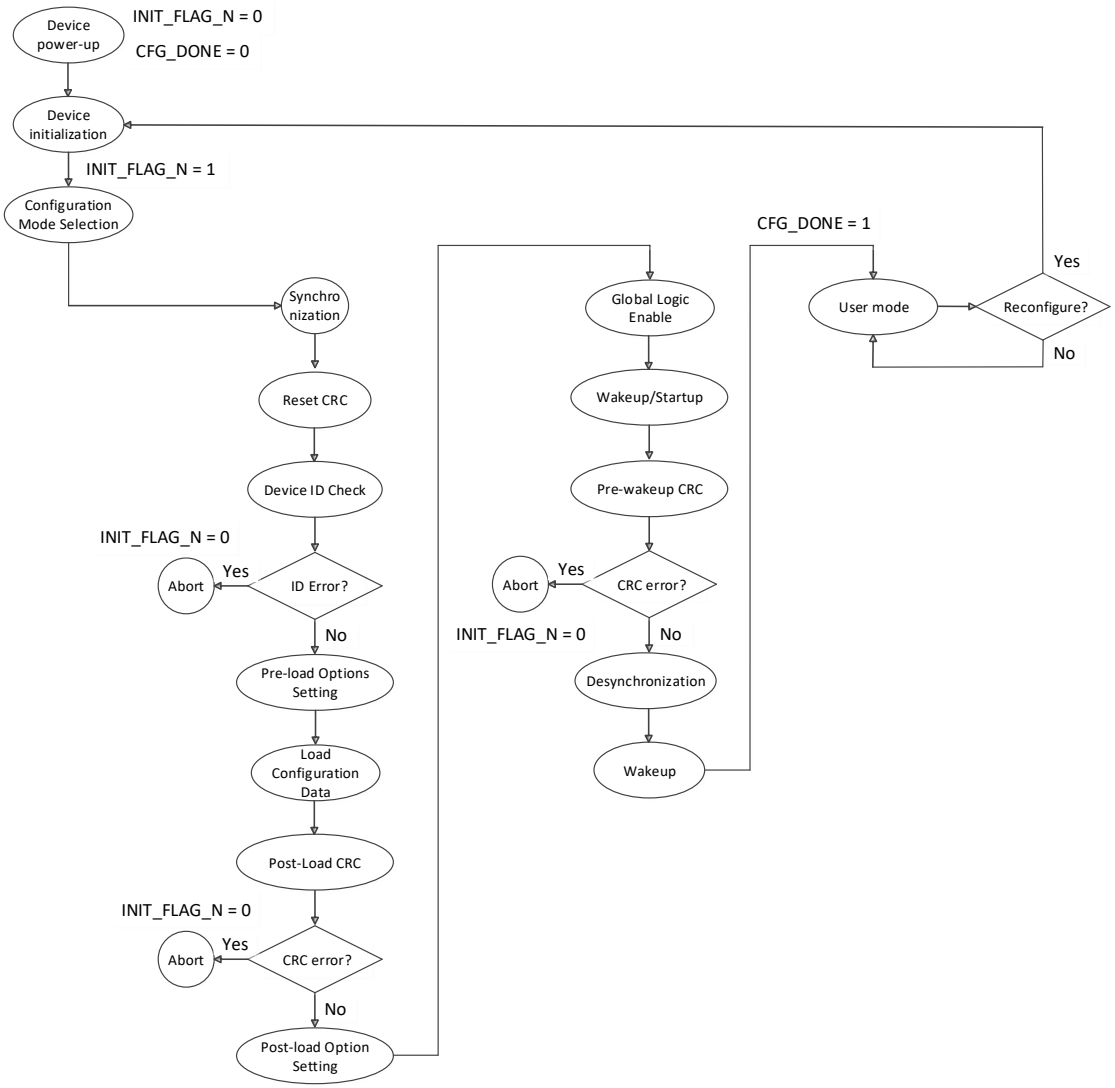

Figure 2-1 Download and Configuration Flowchart

For all configuration interfaces, the basic configuration steps are the same, including setup, bitstream loading, and wakeup. The detailed configuration procedures for CPLD devices are shown in the table below.

Table 2-1 CPLD Configuration Procedures

Setup			**Bitstream Loading**							**Wakeup**				
Device Power-Up	Device Initialization	Configuration Mode Selection	Synchronization	Reset CRC	Device ID Check	Pre-load Options Settings	Load Configuration Data	Post-load CRC	Post-load Option Settings	Enables Global Logics	Wakeup	Pre-wakeup CRC	Desynchronization	Wakeup

The power-up configuration timing for CPLD devices is shown in the following diagram.

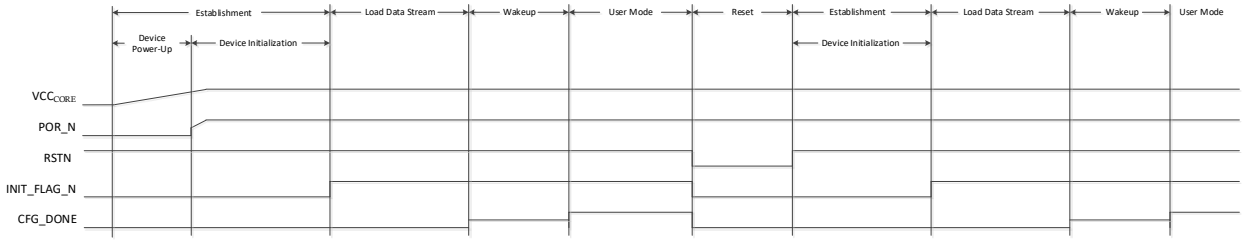

Figure 2-3 Power-Up Configuration Timing

2.1 Setup

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

2.1.1 Device Power-Up

The device power-up process is indicated by the internal POR_N signal. When the POR_N signal goes from low to high, it signifies that the core power VCC_{CORE} has reached the start-up threshold V_{PUP} of the power-up reset circuit.

After the device is powered up, the embedded Flash starts operating normally, followed by a release of the power-up reset of the configuration control system (CCS), which then begins operation.

2.1.2 Device Initialization

After the CCS starts operating, it first initiates the device initialization process.

Device initialization involves the following operations: clearing the configuration memory; powering up the embedded Flash for calibration and adjustment; and reading feature control bits, UID, and embedded Flash lock flag.

Once the device initialization process begins, the CCS starts to empty the configuration memory frame by frame. Upon completion, the initialization complete signal INIT_FLAG_N goes high.

While emptying the configuration memory, the CCS powers up the embedded Flash for calibration and adjustment, and reads feature control bits, UID, and embedded Flash lock flag.

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, Slave SPI instruction reset, Slave I^2C instruction reset or version fallback.

2.1.3 Configuration Mode Selection

After the INIT_FLAG_N signal goes high, the CCS selects the configuration function of the multiplexed pins based on the feature control bits.

2.2 Bitstream Loading

Different configuration modes have different configuration interfaces. After configuration mode selection, the pins corresponding to the modes are set as configuration pins.

2.2.1 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 will be ignored.

2.2.2 Reset CRC

Reset the CRC register.

CRC utilize a 32-bit CRC algorithm and are performed before the wake-up operation. When the configuration data is written to the registers, the 5-bit register address is in the high bits and the 32-bit data is in the low bits. These combined form 37-bit data, which is subject to a CRC calculation.

CRC is performed after data is written to the CRC register. If the data written to the CRC register does not match the calculated CRC result, the INIT_FLAG_N signal is pulled low and the CRC error in the STATUSR register goes high.

2.2.3 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 status register, and the configuration process is exited.

2.2.4 Pre-load Options Settings

Before configuration data is loaded, the operating state of the device can be specified through the PDS interface:

Watchdog Settings

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

Wakeup clock selection

Whether to wait for PLL upon wakeup

Master mode clock frequency setting

For operating operations, please refer to the document "Pango_Design_Suite_User_Guide".

2.2.5 Load Configuration Data

Write the configuration data frames into the configuration memory.

2.2.6 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_FLAG_N signal to go low and the CRC error flag to be stored in the status register.

2.2.7 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

2.3 Wakeup

After the bitstream is loaded and passes the CRC, the CPLD Device enters the wake-up stage. The configuration system first enables the logical outputs of all functional modules inside the CPLD device. 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.

2.3.1 Enables Global Logics

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

2.3.2 Wakeup

The wakeup operation is initiated after global logic is enabled.

2.3.3 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_FLAG_N signal to go low and the CRC error flag to be stored in the status register.

2.3.4 Desynchronization

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

2.3.5 Wakeup

After desynchronization is completed, the wake-up circuit begins to operate according to the set timing, gradually releasing global signals, and finally entering the user mode.

Chapter 3 Embedded Flash

The Pango Compact family CPLD features an on-chip nonvolatile storage unit—embedded Flash. In terms of configuration, it brings great advantages in improving configuration efficiency and reducing peripheral circuits. embedded Flash is a vital component of the Compact family CPLD. This chapter mainly introduces the storage structure information of the embedded Flash and the highly crucial storage content—the detailed description of the feature control bits.

3.1 Storage Structure

The Embedded Flash consists of ordinary memory and supervisory memory. It stores feature control bits, UID, bitstreams and general user data; see the table below for storage locations.

Table 3-1 Embedded Flash Storage Structure

Storage Location	**Storage Content**
Supervisory Memory	Feature Control Bits
	UID
Ordinary Memory	Bitstream or general user data

Embedded Flash indicates the status through the [21:18] bits in the status register. When performing read/write operations on the embedded Flash, apart from reading the UID, it is required to ensure the embedded Flash is in a woken and idle state, i.e., bit [21] busy is 0, bit [19] wake is 1, and bit [18] sleep is 0.

Bit [21] busy indicates whether the operation is complete; an incomplete operation will cause busy to remain high, and only by repowering can the busy bit be reset. Common incomplete operations primarily include not fully reading or writing a page when reading/writing embedded Flash, particularly when writing the last page of a bitstream. The last page should be filled with all 0s or all 1s to complete the page if not full.

The ordinary memory of embedded Flash also supports the lock feature, by which the user can specify the locked ending page through software or other interface instructions. After locking, the contents between page 0 and the specified page (inclusive) cannot be read or written, and can only be unlocked by completely erasing the ordinary memory space. The lock status does not affect the CCS from obtaining the bitstream from the embedded Flash.

The ordinary memory storage space of the embedded Flash in different Compact family CPLD devices varies, as shown in the table below:

Table 3-2 Ordinary Memory Size of Embedded Flash

Device	Number of Heaps	Number of Pages Per Heap	Page Size (Bytes)	Total Capacity (Bytes)	Total Pages of Embedded Flash
PGC1KG/L	1	332	256	84992	332
PGC2KG/L	1	332	256	84992	332
PGC4KD/L	4	320	256	327680	1280
PGC7KD	4	452	256	462848	1808
PGC10KD	4	640	256	655360	2560

Ordinary memory can be used to store bitstreams and user data; when storing user data, care must be taken not to overwrite the bitstream. Refer to the number of pages occupied by the bitstream of each device.

3.2 Feature Control Bits

The feature control bits stored in the embedded Flash determine the specific configuration mode used by the CPLD, the functional settings of different configuration modes, and the reservation settings of reconfigurable pins.

The user can select the configuration mode of the CPLD device by setting different feature control bits.

CPLD devices support JTAG, slave SPI, slave I^2C and the internal APB interface for accessing the feature control bit registers in the embedded Flash, thereby altering the feature control bits. The user can also make modifications through the Fabric Configuration plugin in the PDS software; for the specific procedure, please refer to the "Fabric Configuration User Guide".

During the initialization process, CPLD devices read the feature control bits from the embedded Flash and then select the configuration mode based on these bits, and enable or disable certain configuration interfaces. Therefore, after updating the feature control bits via JTAG, slave SPI, slave I^2C and the internal APB interface, it is necessary to make the CPLD device reread and make the updated feature control bits effective through instruction reset, external RSTN pin reset (the pin should be enabled or reserved), or repowering. All of the above configuration interfaces support "Reset" instructions, see specific instructions at Description of Configuration Modes.

For 1K and 2K devices, 32 feature control bits make up one feature control group CTL, with each feature control bit described as follows.

Table 3-3 1K and 2K Feature Control Group CTL

Item	Bit	Default Value (bin)	Description
cfg_msd_en	[0]	0	Master Self Download mode enable, active high
cfg_rstn_en_n	[1]	1	RSTN pin enable in configuration mode, active low
cfg_init_en	[2]	0	INIT_FLAG_N pin enable in configuration mode, active high
cfg_done_en	[3]	0	CFG_DONE pin enable in configuration mode, active high
cfg_mspi_en	[4]	0	Master SPI mode enable, active high
cfg_jtag_en_n	[5]	0	JTAG interface enable in configuration mode, active low
cfg_sspi_en_n	[6]	0	Slave SPI interface enable in configuration mode, active low
cfg_si2c_en_n	[7]	0	Slave I^2C interface enable in configuration mode, active low
cfg_spal_en_n	[8]	1	Slave parallel interface enable in configuration mode, active low
fallback_en	[9]	0	Dual-boot enable, active high
fastmode	[10]	0	Master SPI fast mode enable, active high
persist_rstn_n	[11]	1	RSTN pin enable in user mode, active low
persist_init	[12]	0	INIT_FLAG_N pin enable in user mode, active high
persist_done	[13]	0	CFG_DONE pin enable in user mode, active high
persist_jtag_n	[14]	0	JTAG interface enable in user mode, active low
persist_sspi_n	[15]	0	Slave SPI interface enable in user mode, active low
persist_si2c_n	[16]	0	Slave I^2C interface enable in user mode, active low
persist_spal_n	[17]	1	Slave parallel interface enable in user mode, active low
i2c_addr[9:0]	[27:18]	1111000000	Chip I^2C slave device 10-bit address
persist_mspi	[28]	0	Master SPI interface enable in user mode, active high
loose_en	[29]	0	Slow read enable 0: Master Self Download, when reading embedded Flash instructions, feature control bit instructions, and trim instructions, The embedded Flash interface uses normal read mode 1: All embedded Flash read operations use slow read mode
Reserved	[31:30]	-	

For 4K/7K/10K devices, a 32-bit feature control bit forms one feature control group CTL, with each feature control bit description as follows.

Table 3-4 4K and 7K Feature Control Group CTL

Item	Bit	Default Value (bin)	Description
cfg_msd_en	[0]	0	Master Self Download mode enable, active high
cfg_rstn_en_n	[1]	1	RSTN pin enable in configuration mode, active low
cfg_init_en	[2]	0	INIT_FLAG_N pin enable in configuration mode, active high
cfg_done_en	[3]	0	CFG_DONE pin enable in configuration mode, active high
cfg_mspi_en	[4]	0	Master SPI mode enable, active high
cfg_jtag_en_n	[5]	0	JTAG interface enable in configuration mode, active low
cfg_sspi_en_n	[6]	0	Slave SPI interface enable in configuration mode, active low
cfg_si2c_en_n	[7]	0	Slave I^2C interface enable in configuration mode, active low

Item	Bit	Default Value (bin)	Description
cfg_spal_en_n	[8]	1	Slave parallel interface enable in configuration mode, active low
fallback_en	[9]	0	Dual-boot enable, active high
fastmode	[10]	0	Master SPI fast mode enable, active high
persist_rstn_n	[11]	1	RSTN pin enable in user mode, active low
persist_init	[12]	0	INIT_FLAG_N pin enable in user mode, active high
persist_done	[13]	0	CFG_DONE pin enable in user mode, active high
persist_jtag_n	[14]	0	JTAG interface enable in user mode, active low
persist_sspi_n	[15]	0	Slave SPI interface enable in user mode, active low
persist_si2c_n	[16]	0	Slave I^2C interface enable in user mode, active low
persist_spal_n	[17]	1	Slave parallel interface enable in user mode, active low
i2c_addr[9:0]	[27:18]	1111000000	Chip I^2C slave device 10-bit address
persist_mspi	[28]	0	Master SPI interface enable in user mode, active high
loose_en	[29]	0	Slow read enable 0: Master Self Download, when reading embedded Flash instructions, feature control bit instructions, and trim instructions, The embedded Flash interface uses normal read mode 1: All embedded Flash read operations use slow read mode
msdboot_en	[30]	0	Master Self Download dual-boot enable, active high
Reserved	[31]	-	

Chapter 4 Description of Configuration Modes

This chapter primarily describes the application interfaces and self-development process for the five configuration modes supported by CPLD devices.

It is particularly important to note that at any one time, operations on the CRAM or embedded Flash can only be performed through one interface among JTAG, slave SPI, slave I^2C, and the internal slave APB interface.

4.1 JTAG Configuration Mode

The JTAG interface is shown in the following figure:

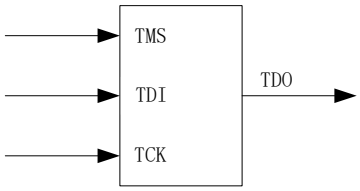

Figure 4-1 JTAG Interface Diagram

The list of JTAG interfaces is as follows:

Table 4-1 JTAG Port List

Item	I/O	Dedicated/ Multi-function	Description
TCK	I	Dedicated	Test Clock. TCK provides the clock for chip test logic and is a dedicated input pin. It can operate independently of the chip system clock or be synchronized with the operating mode clock.
TMS	I	Dedicated	Test Mode Select. Used to control the status switching of the test access port controller state machine on the rising edge of TCK to move test instructions and test data.
TDI	I	Dedicated	Test Data In. Serial input pin. Used to move the test instructions into the instruction register and the test data into the test data register on the rising edge of TCK.
TDO	O	Dedicated	Test Data Out. Serial output pin. During the instruction shift state, it is used to shift the test instructions out from the instruction register.on the falling edge of the TCK signal. During the data shift state, it is used to shift out the test data stored in the test data register that is placed on the scan chain from TDI to TDO on the falling edge of the TCK signal.

Interface timing is illustrated in the figure below.

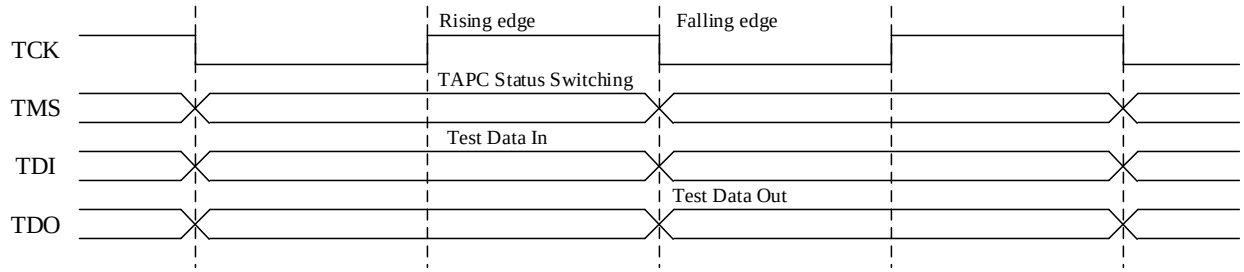

Figure 4-2 JTAG Interface Timing

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

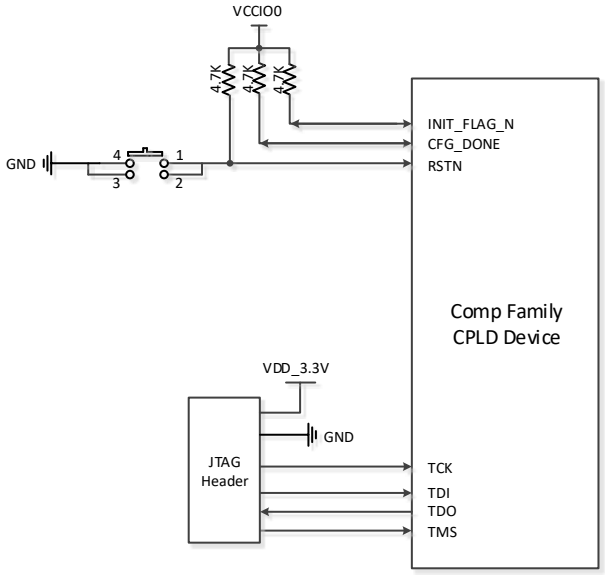

Figure 4-3 JTAG Mode Application Diagram

An RSTN low level will reset the configuration logic; it is recommended to use an external 4.7K resistor to pull it up to VCCIO0. During initialization, if INIT_FLAG_N is used as input and connected to a low level, the CPLD device will remain in the initialization stage. It is advised to use an external 4.7K resistor to pull it up to VCCIO0. When used as an output, INIT_FLAG_N outputs a high level to indicate the end of chip initialization. CFG_DONE outputs a high level to indicate that the chip enters user mode. Inputting an external low level will keep the CPLD device in the configuration stage. It is recommended to use an external 4.7K resistor to pull it up to VCCIO0.

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.

JTAG Cascade Mode

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

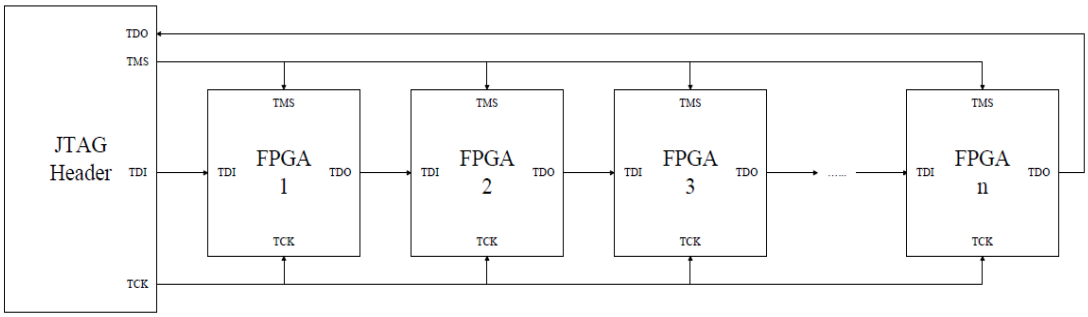

Figure 4-4 JTAG Cascading Application Diagram

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 CPLD device.

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.

4.2 Master SPI Configuration Mode

When applying X1width, the MOSI_SI pin of the CPLD device is connected as command output to the data input of the SPI Flash, and the MISO_SO pin as data input is connected to the data output of the SPI Flash.

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

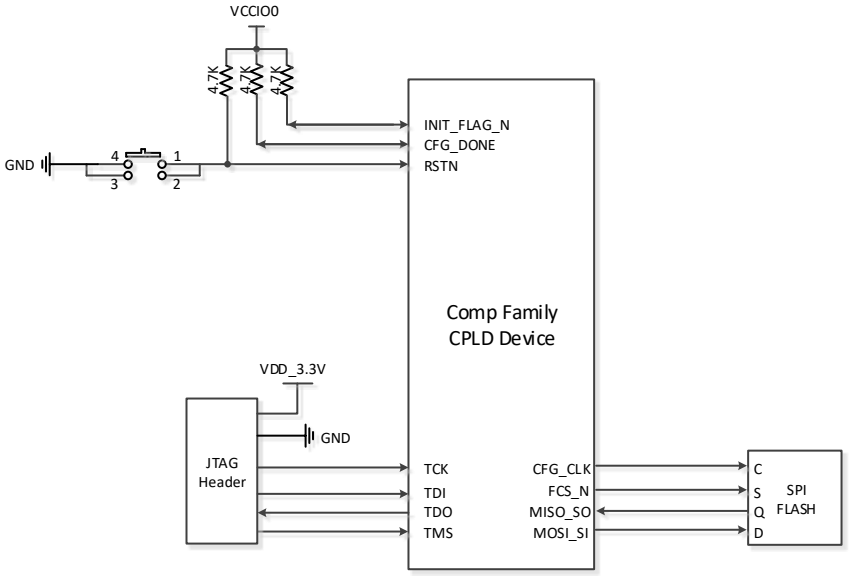

Figure 4-5 Application Diagram of the Master SPI Mode

The interface timing for the Master SPI mode is shown in the diagram below:

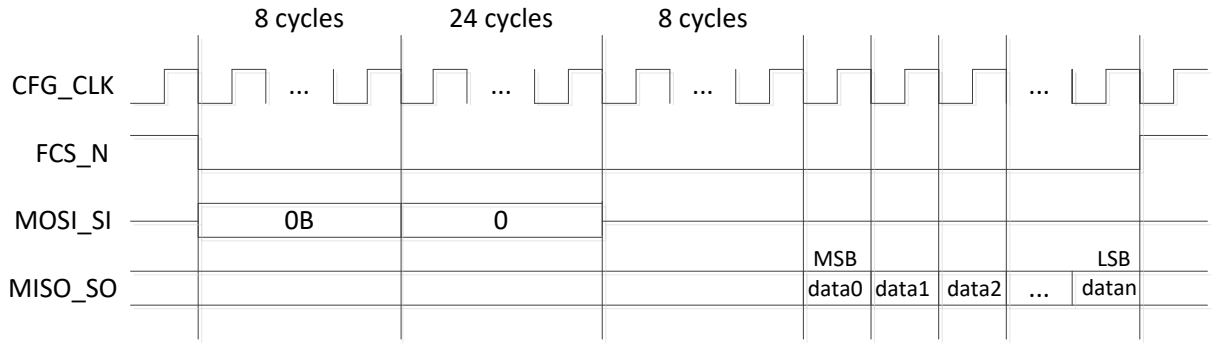

Figure 4-6 Interface Timing Diagram for Master SPI Mode

After the master SPI mode is selected, the CCS will automatically fetch data starting from the 0 address of the SPI Flash by using fast read x1 mode. It is particularly important to note that, to use the Master SPI mode, the Master Self Download should be disabled, i.e., the feature control bit bit[0] cfg_msd_en should be set to 0; otherwise, the Master SPI mode cannot be initiated.

The CPLD device generates the SPI Flash chip select signal at the CFG_CLK falling edge, sending the opcode and address. SPI Flash samples the opcode and address on the rising edge of CFG_CLK. SPI Flash generates data on the falling edge of CFG_CLK, and the CPLD device samples the data on the rising or falling edge of CFG_CLK based on the configuration. 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.

After the specified bitstream is completely retrieved from the SPI Flash, FCS_N is set to 1, ending the operations in master SPI mode.

CPLD devices support direct connection to the 4-pin SPI Flash standard interface, with specific supported models as shown in the table below.

Table 4-2 Supported SPI Flash Models

Company	Model	Density	Bit width	Frequency (Mhz)	Data Delay (ns)	Operating Voltage (V)	Operating Temperature (Celsius)
Micron	M25P05A	512K	X1	50	8	2.3-3.6	-40-85
Micron	M25P10A	1M	X1	50	8	2.3-3.6	-40-125
Micron	M25P20	2M	X1	75	6	2.3-3.6	-40-125
Micron	M25P40	4M	X1	75	6	2.3-3.6	-40-125
Micron	M25PE10	1M	X1	75	6	2.7-3.6	-40-125
Micron	M25PE20	2M	X1	75	6	2.7-3.6	-40-125
Micron	M25PE40	4M	X1	75	6	2.7-3.6	-40-85
Micron	M45PE10	1M	X1	75	8	2.7-3.6	-40-85
Micron	M45PE20	2M	X1	75	8	2.7-3.6	-40-85

Company	Model	Density	Bit width	Frequency (Mhz)	Data Delay (ns)	Operating Voltage (V)	Operating Temperature (Celsius)
Micron	M45PE40	4M	X1	75	8	2.7-3.6	-40-85
Winbond	W25Q20CL	2M	X1/X2/X4	104	8	2.3-3.6	-40-85
Winbond	W25Q40CL	4M	X1/X2/X4	104	8	2.3-3.6	-40-85
Winbond	W25X05CL	512K	X1/X2	104	8	2.3-3.6	-40-85
Winbond	W25X10CL	1M	X1/X2	104	8	2.3-3.6	-40-85
Winbond	W25X20CL	2M	X1/X2	104	8	2.3-3.6	-40-85
Winbond	W25X40CL	4M	X1/X2	104	8	2.3-3.6	-40-85
Winbond	W25X40CV	4M	X1/X2	80	8	2.6-3.6	-40-105
Winbond	W25Q10EW	1M	X1/X2/X4	104	6	1.65-1.95	-40-85
Winbond	W25Q20EW	2M	X1/X2/X4	104	6	1.65-1.95	-40-85
Winbond	W25Q40EW	4M	X1/X2/X4	104	6	1.65-1.95	-40-85
Winbond	W25Q20BW	2M	X1/X2/X4	80	7	1.65-1.95	-40-85
Winbond	W25Q40BW	4M	X1/X2/X4	80	7	1.65-1.95	-40-85
Spansion	S25FL204K	4M	X1/X2	85	10	2.7-3.6	-40-85
GigaDevice	GD25D05B	512K	X1/X2	80	6	2.7-3.6	-40-85
GigaDevice	GD25D10B	1M	X1/X2	80	6	2.7-3.6	-40-85
GigaDevice	GD25Q20C	2M	X1/X2/X4	120	7	2.7-3.6	-40-85
GigaDevice	GD25Q40C	4M	X1/X2/X4	120	7	2.7-3.6	-40-125
GigaDevice	GD25Q21B	2M	X1/X2/X4	104	6	2.7-3.6	-40-85
GigaDevice	GD25Q41B	4M	X1/X2/X4	104	6	2.7-3.6	-40-85
GigaDevice	GD25VQ20C	2M	X1/X2/X4	104	7	2.3-3.6	-40-85
GigaDevice	GD25VQ40C	4M	X1/X2/X4	104	7	2.3-3.6	-40-85
GigaDevice	GD25VQ21B	2M	X1/X2/X4	104	6	2.3-3.6	-40-85
GigaDevice	GD25LQ05B	512K	X1/X2/X4	104	7	1.65-2.1	-40-85
GigaDevice	GD25LQ10B	1M	X1/X2/X4	104	7	1.65-2.1	-40-85
GigaDevice	GD25LQ20B	2M	X1/X2/X4	104	7	1.65-2.1	-40-85
GigaDevice	GD25LQ40B	4M	X1/X2/X4	104	7	1.65-2.1	-40-85
GigaDevice	GD25LQ40	4M	X1/X2/X4	120	7	1.65-1.95	-40-85

Note: The models listed in the table have been tested. Other models should undergo testing and verification before use.

4.3 Master Self Download Configuration Mode

The Master Self Download configuration mode is designed for embedded Flash. First, set the feature control bit to enable the Master Self Download mode, that is, bit [0] cfg_msd_en. Then, the bitstream is loaded into the embedded Flash, and upon resetting the device, CCS will automatically read the bitstream from the embedded Flash to configure the CPLD.

4.4 Slave SPI Configuration Mode

Pango Compact family CPLDs provide the slave SPI interface as one of the access interfaces for the chip CCS. With the slave SPI interface, users can configure/reconfigure the CRAM, program embedded FLASH, read/write feature control bits, etc.

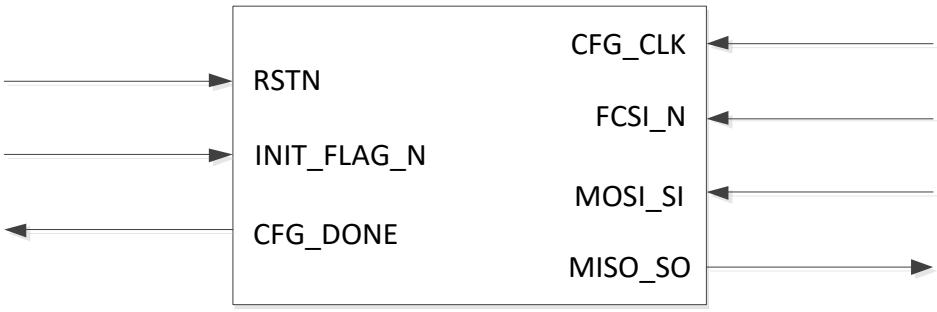

Figure 4-7 Slave SPI Interface Diagram

The list of ports is shown in the table below.

Table 4-3 List of SPI Ports

Item	I/O	Description
RSTN	I	Reset pins; active-low Perform an asynchronous reset on the entire chip, apart from the JTAG TAP controller. When it remains at a low level, the chip cannot be configured. INIT_FLAG_N is at a low level simultaneously. A low level triggers a reset and returns the chip configuration to its initial state. The configuration storage clearing and the configuration process are restarted. When this pin is floating, it is recommended that the pin be fixed in a weak pull-up state.
CFG_CLK	I	Input clock, up to 100 MHz. When "READ" and "READ_CTL" instructions are executed, the transmission time for pad bytes must not be less than 800ns. It can be done in multiple ways: 1. Make clock output constant. When "READ" and "READ_CTL" instructions are executed, the maximum clock frequency is 10 MHz. 2. Keep the maximum clock frequency at 100 MHz. The clock is discontinuous during the transmission of pad bytes. After transmitting the first half of a pad byte, the clock pauses for no less than 800ns, then continues to transmit the second half of the pad byte. After the CPLD device enters user mode, if the slave SPI interface is still used as a configuration pin, it keeps as such; if the SPI interface is fully released for user use, it is used as such.
INIT_FLAG_N	Open-drain	A signal indicating the completion of CPLD device initialization or a configuration error After the chip is powered up or reset, this pin becomes a bidirectional interface for driving a low level internally by the internal circuit and inputting to the chip simultaneously. Upon the completion of the CPLD device initialization, the pin goes high (externally pulled up), and when the pin is high, the mode pin is sampled. This pin is set to open-drain output and is pulled high by an external pull-up resistor. During configuration, this pin is bidirectional. If an error occurs, the pin is driven by the internal circuit to output a low level while feedbacking the pin's state to the inside of the chip.

Item	I/O	Description
		After CPLD device initialization is completed, this pin can continue to be driven externally to a low level to delay configuration until the pin goes from low to high (must be set to open-drain output). After the CPLD device enters user mode, this pin can continue to be used as a configuration pin or be released for user use.
CFG_DONE	Open-drain	Indicate configuration completion. 0: CPLD device not configured 1: CPLD device configured
FCSI_N	I	Chip select input pin for the slave SPI interface, active low. After the CPLD device enters user mode, if the slave SPI interface is still used as a configuration pin, it keeps as such; if the SPI interface is fully released for user use, it is used as such.
MOSI_SI	I	Pin for SPI data bus master device output and slave device input. After the CPLD device enters user mode, if both the master SPI interface and the slave SPI interface are released for user use, the MOSI_SI pin is released for user use.
MISO_SO	O	Pin for SPI data bus master device input and slave device output After the CPLD device enters user mode, if both the master SPI interface and the slave SPI interface are released for user use, the MISO_SO pin is released for user use.

The slave SPI interface provides a convenient and universal method for user configuration. It is easier to implement than the JTAG interface and features a higher transmission rate than the IIC interface. Additionally, it provides more flexible version management capabilities and a wider range of chip operations than Master Self Download and master SPI modes. Typically, devices such as CPU, MCU, DSP, etc., can act as a master while the CPLD as a slave. This application can be illustrated by the following figure:

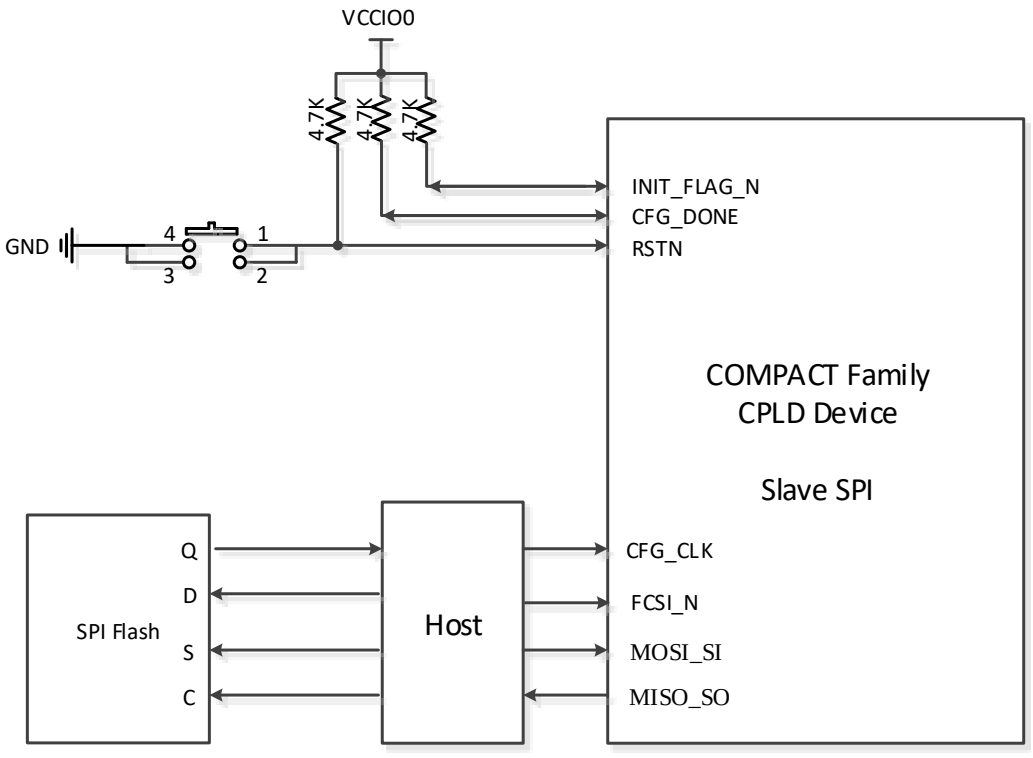

Figure 4-8 Application Diagram of the Slave SPI

Pango provides a C code implementation for the slave SPI operational procedures. Refer to the application guide "AN03009_Compact Family CPLD Device Slave SPI Interface Configuration and Programming Guide", as well as the corresponding example project.

For the usage instructions of the slave SPI interface, see Chapter 7 Configuration Details. It is particularly important to note that FCSI_N must be pulled high before and after the execution of any instructions, and at least 8 clock cycles must be provided.

4.5 Slave I^{2}C Configuration Mode

The Pango Compact family CPLD provides a slave I^{2}C interface as one of the access interfaces for the chip CCS. The slave I^{2}C mode interface supports fast mode, allowing users to configure/reconfigure CRAM, program embedded Flash, read and write feature control bits, etc. through the slave SPI interface.

Figure 4-9 Slave I^{2}C Port Diagram

The list of ports for the slave I^{2}C interface is as follows:

Table 4-4 Slave I^{2}C Port List

Item	I/O	Description
SCL	I	Serial clock, with a maximum frequency of 400KHz
SDA	IO	Serial Data

The slave I^{2}C interface uses fewer IOs, offering the advantage of saving hardware resources. Typically, devices such as CPU, MCU, DSP, etc., can act as a master while the CPLD as a slave. This application can be illustrated by the following figure:

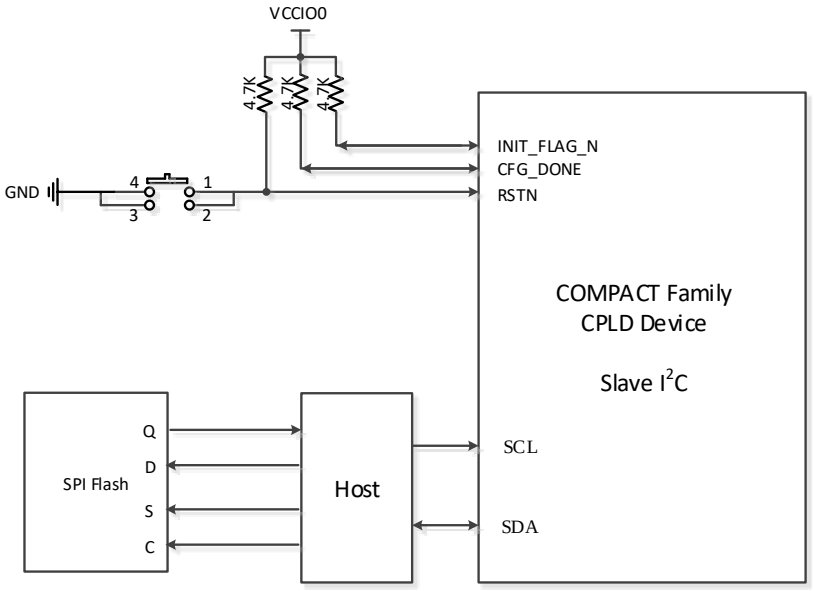

Figure 4-10 Slave I^{2}C Application Diagram

Pango provides a C code implementation for the slave I^{2}C operational procedures. Refer to the "AN03011_Compact Family CPLD Device Slave I2C Interface Configuration and Programming Guide", as well as the corresponding example project.

Note that if users wish to operate the device via slave I^{2}C in user mode, the I^{2}C reservation bit of the feature control bit, namely bit [16]persist_si2c_n, should be reserved to 0. The automatic shut off of the OSC shall be deselected; correspondingly the [Project Setting > Generate Bitstream > Configuration > Enable OSC Shut Off] shall be deselected in the PDS settings.

Chapter 5 Dual Boot

All CPLD devices support dual boot functionality, but only type D devices support master self download dual-boot functionality. In regular dual boot applications, two bitstreams are stored in the embedded Flash and external SPI Flash, respectively. Based on the feature control bit settings, the embedded Flash or SPI Flash is prioritized for loading the application bitstream. If the application bitstream loading fails, the golden bitstream is then loaded from another storage location. In the master self download dual-boot applications, both sets of bitstreams are stored in embedded Flash, with the golden bitstream starting at address 0 and the application bitstream immediately following the golden bitstream on the next page. After a failure in loading the application bitstream, the device automatically falls back to the golden bitstream at address 0.

The PGC family devices that support the dual-boot feature are listed in the following table:

Table 5-1 Devices that Support Dual-Boot

Device	**Regular Dual Boot**	**Master Self Configuration Dual Boot**
PGC1KG/L	Supported	Not supported
PGC2KG/L	Supported	Not supported
PGC4KD	Supported	Supported
PGC4KL	Supported	Not supported
PGC7KD	Supported	Supported
PGC10KD	Supported	Supported

During dual-boot, the watchdog must be enabled (to detect timeouts during configuration and in user mode; the watchdog counter decrements by 1 every 512 system clock cycles; for PGC10KD, the system clock is 15.647MHz, which is about 32.7us, decrementing the watchdog counter by 1; for other devices, the system clock is 20.46MHz, which is about 25us, decrementing the watchdog counter by 1). If an error occurs during the loading of the current application bitstream, the device can re-download the golden bitstream; if there is also an error in the configuration of the golden bitstream, there will no longer be a reset, instead, INIT_FLAG_N will be set to 0, ending the dual-boot operation and disabling the watchdog.

During the device loading process, the following errors will trigger a fallback:

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

To determine if the currently loaded bitstream has errors, read the status register's timeout, crc_err, and id_err. After fallback, the error signals in the status register are cleared, and the fallback in the status register indicates whether the current bitstream is the fallback bitstream.

The process of master self download dual-boot is as follows.

1. Load the golden bitstream from embedded Flash address 0.
2. The master self download dual-boot command in the golden bitstream causes the application bitstream to be loaded from the embedded Flash address specified by DBOOTADRR.
3. If there is an error during loading the application bitstream, the golden bitstream will be loaded from embedded Flash address 0.
4. If there is also an error during the configuration of the golden bitstream, there will no longer be a reset; instead, INIT_FLAG_N is set to 0, ending the master self download dual-boot operation.
5. After successful configuration of the application bitstream or golden bitstream, user mode is used.
6. In user mode, the application bitstream and user Flash content can be updated through the internal slave APB interface, JTAG interface, slave SPI interface, or slave I^2C interface.
7. After the application bitstream and user Flash content are updated, a reset can be performed through the internal slave APB interface, JTAG interface, slave SPI interface, or slave I^2C interface to achieve an upgrade.

5.1 Operation Flow

For non-master self download dual-boot, just enable dual-boot by setting bit [9] fallback_en of the feature control bits to 1. The bitstream in the embedded Flash and the external SPI Flash should be consistent with that of the master self download and the master SPI mode bitstreams. If the priority is to load from embedded Flash, i.e., to use the bitstream from embedded Flash as the application bitstream, then set the bit [0] cfg_msd_en of the feature control bits to 1 and the bit [4]cfg_mspi_en to 0; if the priority is to load from SPI Flash, i.e., to use the bitstream from SPI Flash as the application bitstream, then set the bit [0] cfg_msd_en of the feature control bits to 0 and bit [4]cfg_mspi_en to 1.

For the master self download dual-boot, it is necessary to simultaneously activate the master self download, dual-boot enable, and master self download dual-boot enable by setting bits [0] cfg_msd_en, [9] fallback_en, and [30] msdboot_en of the feature control bits to 1.

The master self download dual-boot bitstream can be generated using the PDS download tool. First, generate two sfc files, one for the golden bitstream and the other for the application bitstream, with the Flash device settings remaining as default, as shown in the figure below:

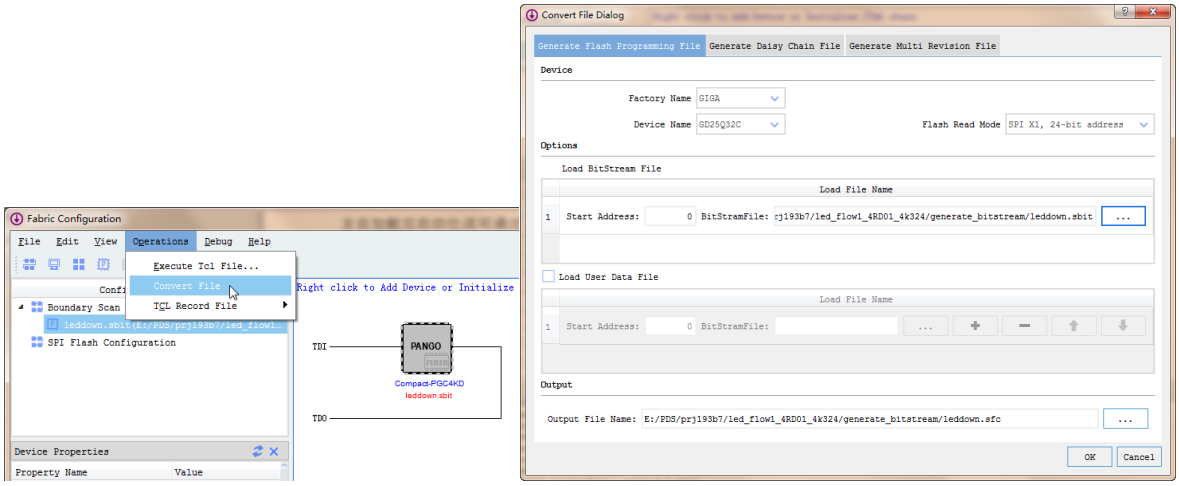

Figure 5-1 Generating Golden Bitstream or Application Bitstream SFC

Then, use the conversion tool to create the master self download dual-boot bitstream, with the one at the first bitstream position serving as the golden bitstream. This is shown in the following figure:

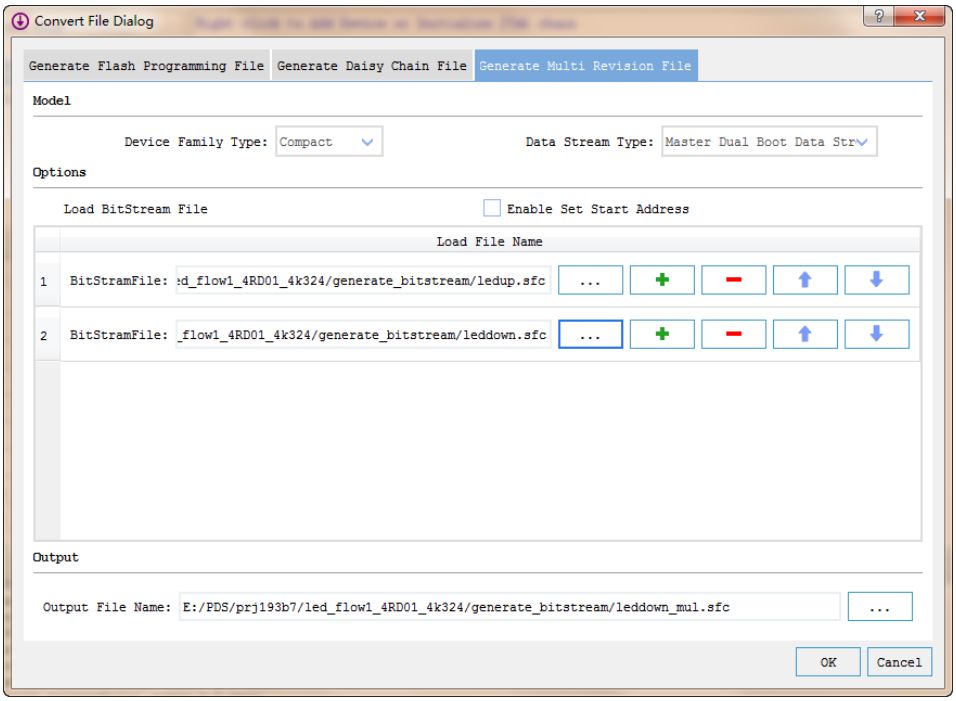

Figure 5-2 Generating Master Self Download Dual-Boot Bitstream

During synthesis, it is possible to set the bitstream starting address, with the addresses of various sections generated under the default setting as shown in the following table:

Table 5-2 Default Storage Intervals for Dual-Boot Bitstreams

Device	Master Self Configuration Dual Boot bitstream size (Kbyte)	Golden bitstream page range	Application bitstream page range	User data page range
PGC4KD	285.5	0~570	571~1141	1142~1279
PGC7KD	445.5	0~890	891~1781	1782~1807
PGC10KD	638.75	0~1277	1278~2554	2555~2559

By default, the sbit file for configuring the CPLD is used when programming the embedded Flash. However, the generated files are sfc files, and to download them into the embedded Flash, it is necessary to modify the files used for programming the embedded Flash in the [Set Device Properties]. This is shown in the following figure:

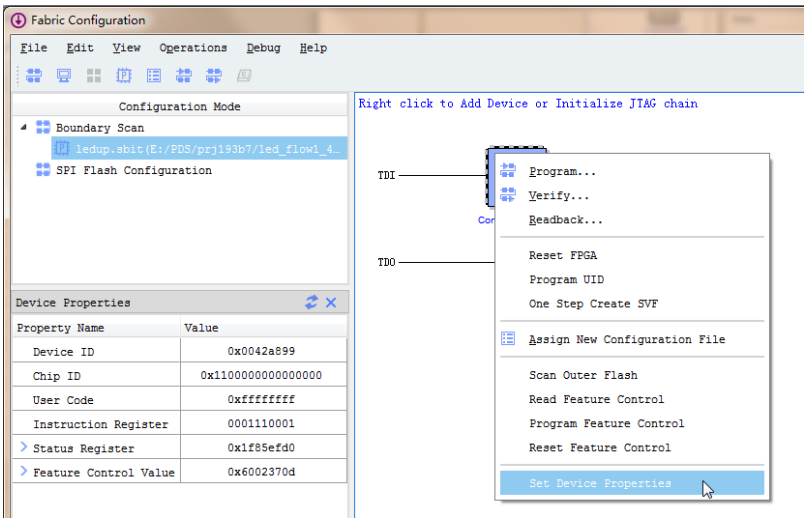

Figure 5-3 Modify Device Properties

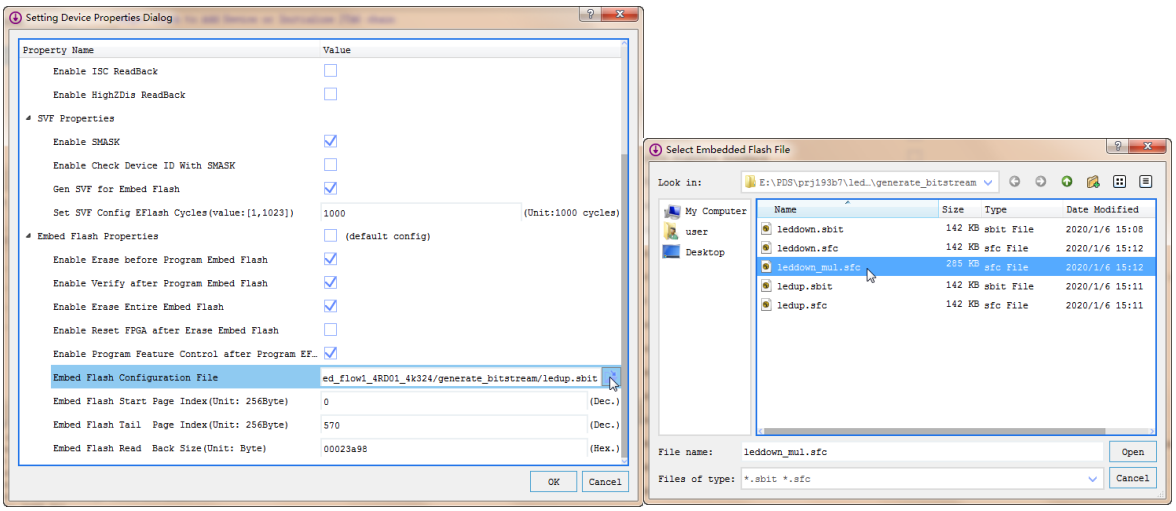

Figure 5-4 Selecting the Master Self Download Dual-Boot Bitstream SFC File

Then, program the file into the embedded Flash. For specific operations on updating application bitstreams and user data in user mode, please refer to Chapter 7 Configuration Details and Chapter 8 Boundary Scan and JTAG Configuration.

After updating the embedded Flash, send the "RESET" instruction to upgrade the bitstream in the configuration memory. Reset options include internal reset salve APB interface reset, JTAG interface reset, slave SPI interface reset, slave I^2C interface reset, and RSTN pin hard reset (which requires setting the feature control bits to preserve the RSTN pin).

Chapter 6 PDS Software Download Configuration

CPLD devices integrate all operations related to JTAG and master SPI configuration into the PDS software. The PDS design software will generate configuration files with different extensions to accommodate various configuration schemes as detailed in the table below.

Table 6-1 Descriptions of PDS Configuration Files

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

The .sbit file is the default bitstream file generated upon the compilation of the PDS tool and can be generated by running "Generate Bitstream". For specific operational procedures, refer to the "Design Editor User Guide" and "Fabric Configuration User Guide".

CPLD devices support both normal bitstreams and compressed bitstreams. The compressed bitstream file is smaller than those before compression, and the size of the compressed file depends on the design.

Table 6-2 Bitstream sizes for Each Device in the PGC Family

Compact Family CPLD Devices	File Name	Regular File Size (KB)	Number of embedded Flash Pages Occupied by Normal Bitstreams
1KL	*.sbit	52.6	211
1KG	*.sbit	82.9	332
2KL	*.sbit	82.9	332
2KG	*.sbit	82.9	332
4KL	*.sbit	142.7	571
4KD	*.sbit	142.7	571
7KD	*.sbit	222.6	891
10KD	*.sbit	319.2	1277

Chapter 7 Configuration Details

7.1 Packet Types

The bitstream consists of multiple packets, with each packet comprising a header and a data section. The header is a 32-bit word. Following the header is the data. 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 packet format is shown as follows:

Table 7-1 Packet Format

Packet Header Type [31:29]	**Opcode [28:27]**	**Register Address [26:22]**	**Number of 32-Bit Words [21:0]**
101	00: No operation 01: Write 10: Read 11: Reserved	Register address	The number of 32-bit words following the header

7.2 Configuration register

The CCS system of CPLD devices includes the following registers, and the register addresses should be written starting with the high bits.

Table 7-2 List of Configuration Memory Addresses

Item	**R/W**	**Address**	**Description**
CRCR	R/W	00000	CRC Registers
CMDR	R/W	00010	Command register
CTRLR	R/W	00100	Control Register
MFWRITER	W	00110	Multi-frame write register
STATUSR	R	01001	Status register
DBOOTADRR	R/W	10000	Dual-Boot Address Registers
WATCHDOGR	R/W	10001	Watchdog status register
CMASKR	R/W	10111	Control master register
RBCRCR	R/W	11110	Readback CRC Registers

7.2.1 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_FLAG_N (init_complete) is pulled low.

7.2.2 Command Register (CMDR)

The command register determines the next operation to be performed by the CCS.

Table 7-3 Command Register Instruction Descriptions

Command	Code	Description
NOP	00000	No Operation (Initial Value)
RSTCRC	00001	Reset CRC register
WCMEM	00100	Write configuration data, used before writing the configuration data of the configuration memory through CMEMIR
MFWRITE	00101	Multi-frame write, for writing the current frame of data into subsequent continuous frames. The number of frames is determined by MFWRITER
RCMEM	00110	Read configuration data, used before back-reading the configuration data of 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	To desynchronize, used at the end of configuration
RWD	01100	Restart the watchdog
WCMEMEN	10000	Write configuration memory enable, used before writing configuration data to enable configuration memory write
WCMEMDIS	10001	Write configuration memory disable, used after writing configuration data to disable configuration memory write
RCMEMEN	10010	Read configuration memory enable, used before reading configuration data to enable configuration memory read
RCMEMDIS	10011	Read configuration memory disable, used after reading configuration data to disable configuration memory read
MSDBOOT	10101	Master Self Download dual-boot, for loading the application bitstream from the address specified in the dual-boot address register

7.2.3 Control Registers (CTRLR)

Table 7-4 Description of Control Registers

Item	Bit	Initial Value	Description
Reserved	[31:5]		
mask_en	[4]	1'b0	Variable storage unit readback mask enable 0 = Mask disabled 1 = Mask enabled No such feature for 2K Devices
mfg_sram_retention	[3]	1'b0	When set to 1 (and gwen=1), enables SRAM retention voltage test (Domain is 1.2V)
osc_off_en	[2]	1'b1	Control whether CCS logic can turn off OSC 0: Not allow 1: Allow

Item	Bit	Initial Value	Description
wrctrl	[1:0]	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: Reconfiguration enable; backread enable (default) 01: Reconfiguration enabled, readback disabled 1x: Reconfiguration disabled, readback disabled

7.2.4 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.

7.2.5 Status Register (STATUSR)

The configuration pin status and the operating status of the chip are recorded in the status register and can be read via JTAG, slave SPI, slave I^2C, and internal slave APB interface. A detailed description is shown in the table below.

Table 7-5 Status Register (STATUSR)

Item	Bit	Description
Reserved	[31:30]	
persist_mspi	[29]	Master SPI interface enable in user mode, active high
persist_si2c	[28]	Slave I^2C interface enable in user mode, active high
persist_sspi	[27]	SPI interface enable in user mode, active high
persist_jtag	[26]	JTAG interface enable in user mode, active high
persist_done	[25]	CFG_DONE pin enable in user mode, active high
persist_init	[24]	INIT_FLAG_N pin enable in user mode, active high
persist_rstn	[23]	RSTN pin enable in user mode, active high
pass_cal	[22]	Embedded Flash calibration successful flag, active high For 4K/7K devices, this bit is reserved
busy	[21]	Embedded Flash busy flag, active high
lock	[20]	Embedded Flash lock flag, active high Once embedded Flash is locked, it cannot be programmed, the bitstream data and user Flash cannot be read, and only the Flash can be erased entirely
wake	[19]	Embedded Flash wake-up flag, active high
sleep	[18]	Embedded Flash sleep flag, active high
fallback	[17]	Fallback indication flag, active high
pll_lock	[16]	PLL lock flag, active high

Item	Bit	Description
gwen	[15]	Global write enable, active high
grs_n	[14]	Global register set/reset, active high
gouten	[13]	Global IO output enable, active high
Reserved	[12]	
glogen_fb	[11]	Global logic enable feedback
glogen	[10]	Global logic enable, active high
done_i	[9]	DONE pin input
done	[8]	Device wake-up success flag, active high
init_n	[7]	INIT_FLAG_N pin input
init_complete	[6]	Initialization complete and configuration error indication, active high
wakedown_over	[5]	Wake-up shutdown complete, active high
wakeup_over	[4]	Wake-up complete, active high
timeout	[3]	Watchdog timeout, active high
rbcrc_err	[2]	Readback CRC results 0: CRC correct 1: CRC error
crc_err	[1]	CRC results 0: CRC correct 1: CRC error
id_err	[0]	ID detection results 0: Correct 1: Wrong

7.2.6 Dual-boot address register (DBOOTADRR)

7.2.6.1 1K/2K/4K/7K

Table 7-6 Dual-boot Address Register Description (1)

Item	Bit	Description
Reserved	[31:19]	
Ba[1:0]	[18:17]	Heap Address
Ra[8:0]	[16:8]	Page Address
Reserved	[7:6]	
Ca[5:0]	[5:0]	32-bit data page internal offset address

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

7.2.6.2 10K

Table 7-7 Dual-boot Address Register Description (2)

Item	Bit	Description
Reserved	[31:20]	
Ba[1:0]	[19:18]	Heap Address

Item	Bit	Description
Ra[9:0]	[17:8]	Page Address
Reserved	[7:6]	
Ca[5:0]	[5:0]	32-bit data page internal offset address

7.2.7 Watchdog Register (WATCHDOGR)

For timeout detection in configuration mode.

Table 7-8 Watchdog Register Description

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

7.2.8 Control Mask Register (CMASKR)

The corresponding bits in mask control register, 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 the control register can be written to, while the other bits cannot.

7.2.9 Option Register 0 (OPTION0R)

Table 7-9 Description of Option Register

Item	Bit	Initial Value	Description
Reserved	[31]		
oscfsel_rbcrc	[30:24]	7'd0	Readback CRC clock frequency selection 7'b0000000: 266/128, 2.08MHz 7'b0000001: 266/2, 133.00MHz 7'b0000010: 266/2, 133.00MHz 7'b0000011: 266/3, 88.67MHz 7'b0000100: 266/4, 66.50MHz … 7'b1111110: 266/126, 2.11MHz 7'b1111111: 266/127, 2.09MHz
done_syn	[23]	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 wake-up 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 1: Synchronize
Reserved	[22:12]		

Item	Bit	Initial Value	Description
wait_pll	[11]	1'b0	Wait for PLL lock to be enabled during wakeup 0: No need to wait for PLL lock 1: Wait for PLL lock
startup_sel	[10:8]	3'b000	Wakeup clock selection 000: mclk 001: sclk 010: tck 011: scl 100: uclk 101: spal_clk For the wakeup clock, only the clock used during configuration or the user clock can be selected
Reserved	[7]		
oscfsel	[6:0]	7'd0	Master SPI clock MCLK frequency selection 7'b0000000: 266/128, 2.08MHz 7'b0000001: 266/2, 133.00MHz 7'b0000010: 266/2, 133.00MHz 7'b0000011: 266/3, 88.67MHz 7'b0000100: 266/4, 66.50MHz … 7'b1111110: 266/126, 2.11MHz 7'b1111111: 266/127, 2.09MHz

7.2.10 Readback CRC register (RBCRCR)

CRC value for non-DRM data in the configuration memory.

7.2.11 Device ID Register (IDR)

Device ID. The data stream includes writing the "IDCODE" to the ID register. 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.

Table 7-10 IDR Registers

Device Model	Device ID (32 bits)
1KG	X0499899
1KL	X0410899
2KG	X0419899
2KL	X0411899
4KD	X042A899
4KL	X0422899
7KD	X042B899
10KD	X042C899

7.3 Bitstream Formats

Table 7-11 Bitstream Formats

Content	Description
FFFFFFFF	Padding words (10)
……	
FFFFFFFF	
01332D94	Synchronization
A8800001	Packet Header: Write to CMDR Register
00000000	Data: NOP command
A8800001	Packet Header: Write to CMDR Register
00000001	Data: RSTCRC command
A8400001	Packet Header: Write to IDR Register
xxxxxxxx	Data: IDCODE
AC400001	Packet Header: Write to WATCHDOG Register
xxxxxxxx	Data: WATCHDOG
AE400001	Packet Header: Write to OPTION0R Register
xxxxxxxx	Data: Contents of OPTION0R
AE800001	Packet Header: Write to OPTION1R Register
xxxxxxxx	Data: Contents of OPTION1R
AF800001	Packet Header: Write to RBCRCR Register
xxxxxxxx	Data: Content of RBCRCR
A8800001	Packet Header: Write to CMDR Register
00000002	Data: SWITCH command
A0000000	NOP Packet Header (10)
……	
A0000000	
A8800001	Packet Header: Write to CMDR Register
00000014	Data: "SWITCHRBCRC" Command
A0000000	NOP Packet Header (10)
……	
A0000000	
AF400001	Packet Header: Write to FRAMER Register
xxxxxxxx	Data: Content of FRAMER
A8800001	Packet Header: Write to CMDR Register
00000010	Data: "WCMEMEN" Command
AAC00001	Packet Header: Write to ADRR Register
xxxxxxxx	Data: Contents of ADRR
A8800001	Packet Header: Write to CMDR Register
00000004	Data: WCMEM command
{3'b101, 2'b01, 5'b00101, 22'dx}	Packet Header: Write to CMEMIR Register

Content	Description
xxxxxxxx	Data: Contents written to configuration memory
……	
xxxxxxxx	
A0000000	NOP Packet Header (30)
……	
A0000000	
A8800001	Packet Header: Write to CMDR Register
00000011	Data: "WCMEMDIS" Command
A8000001	Packet Header: Write to CRCR Register
xxxxxxxx	Data: CRC value
ADC00001	Packet Header: Write to CMASKR Register
xxxxxxxx	Data: Contents of CMASKR
A9000001	Packet Header: Write to CTRLR Register
xxxxxxxx	Data: Content of CTRLR
A8800001	Packet Header: Write to CMDR Register
00000009	Data: GUP command
A8800001	Packet Header: Write to CMDR Register
00000007	Data: SWAKEUP command
A8000001	Packet Header: Write to CRCR Register
xxxxxxxx	Data: CRC value
A8800001	Packet Header: Write to CMDR Register
0000000B	Data: DESYNC command
A0000000	NOP Packet Header (100)
……	
A0000000	

7.4 Slave SPI/Slave I^{2}C Instructions Set

Table 7-12 SPI/I^{2}C Instructions Set

Instruction	Description	Op Code
NOP	No Operation	FF
RDID	Read Identification	A1
RDUSER	Read Usercode	A2
RDSR	Read Status Register	A3
RDUID	Read Unique Identification	A4
RDLOCK	Read Embedded Flash Lock Information	A5
CFG	Config Bitstream	50
WREN	Write Enable	51
WRDIS	Write Disable	52

Instruction	Description	Op Code
RESET	Reset CPLD	60
ERASE	Erase Bulk	10
ERASE_PAGE	Erase Page	11
ERASE_CTL	Erase Feature Control bit	12
PROGRAM	Program Page	20
PROGRAM_CTL	Program Feature Control bit	22
READ	Read	30
READ_CTL	Read Feature Control bit	31
PROGRAM_LOCK	Lock Embedded Flash	40
EFlash_SLEEP	Embedded Flash Sleep	70
EFlash_WAKEUP	Embedded Flash Wake Up	71
ISC_ENABLE	ISC Enable After the "ISC_ENABLE" instruction is loaded, the device first samples the output value of the pins to the boundary scan register, then hands over control of the output pins to the boundary scan register	53
ISC_DISABLE	ISC Disable	54
ISC_ENABLE_MSPI	ISC Enable,MSPI reserved After the "ISC_ENABLE" instruction is loaded, the device first samples the output value of the pins to the boundary scan register, then hands control of the output pins to the boundary scan register (only support Slave I^2C interface)	55
ISC_DISABLE_MSPI	ISC Disable,MSPI reserved (only support Slave I^2C interface)	56

Note: For detailed description of the slave SPI/slave I^2C instruction timing, refer to Appendix 4 and Appendix 5. It is particularly important to note that, during operations on the slave SPI interface, FCSI_N must be pulled high before and after the execution of any instructions, and at least 8 clock cycles must be provided.

7.5 Configuring CRAM (Configuration Memory) through Slave SPI/ Slave I^2C Interface

Configuration/Reconfiguration

Configuration refers to the process of configuring a CPLD chip from the start until it enters user mode; Reconfiguration refers to the process of resetting (instruction reset or externally triggered RSTN hard reset), clearing all CRAM, exiting user mode, and then reconfiguring after the CPLD chip enters user mode. The flow is as follows:

1. Power-up
2. When the value of INIT_FLAG_N becomes 1, write the RDID instruction to read the device IDCODE
3. Check the value of IDCODE
4. If IDCODE does not match, terminate the operation. If IDCODE matches, write the WREN instruction

5. Write the CFG instruction to load the bitstream
6. After the bitstream transmission is completed, if the salve SPI/slave I^2C interface is released for user use in user mode (with feature control bit persist_sspi_n/persist_si2c_n set to 1), check the values of INIT_FLAG_N and CFG_DONE to conclude the operation. If the slave SPI interface is used as a configuration interface in user mode, write the WRDIS instruction
7. Write the RDSR instruction to read the device status register

The CPLD device can be reset (by setting RSTN to 0) at any time after power-up. After reset, users can proceed to step 2 to reconfigure the CPLD device.

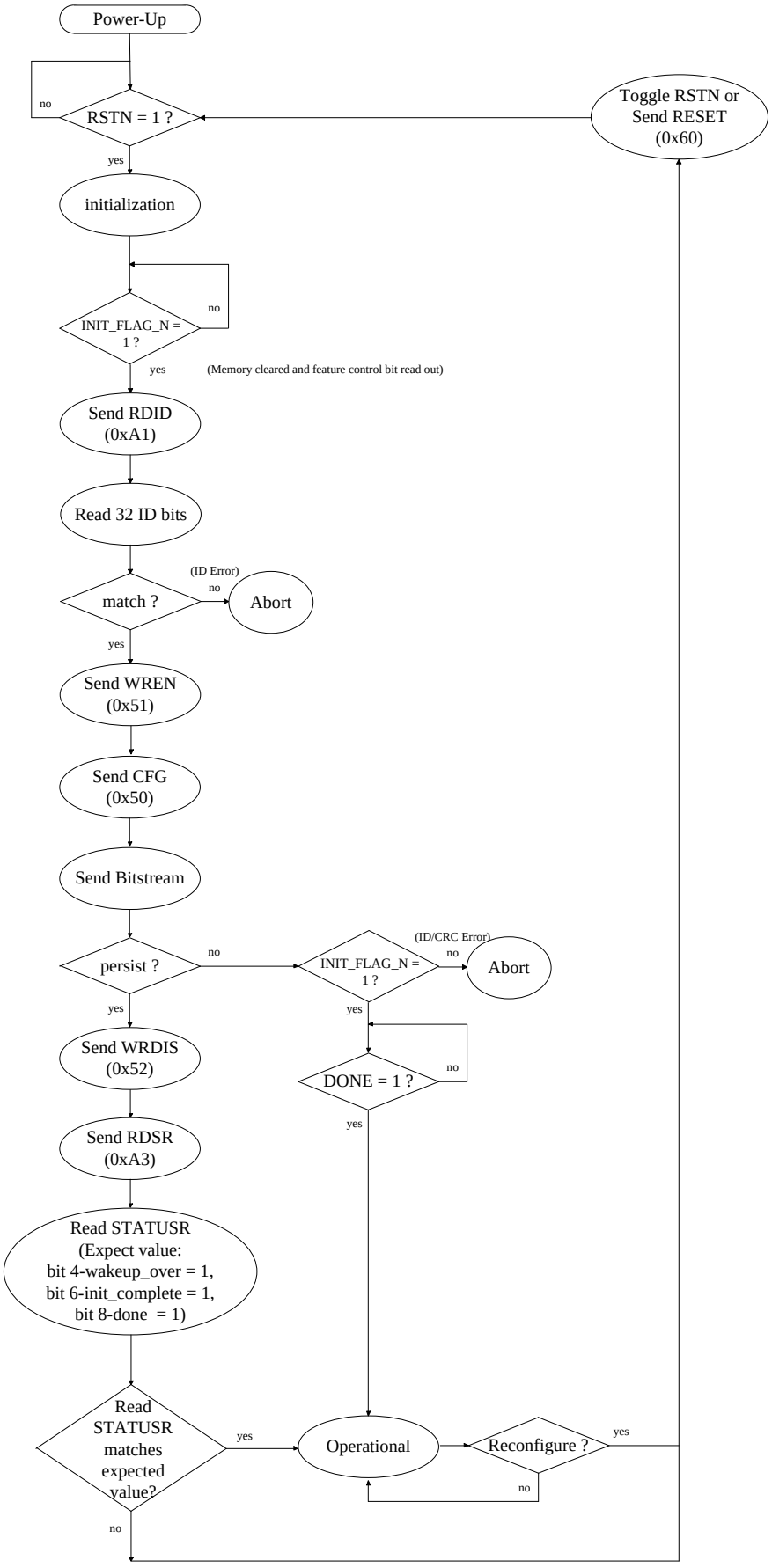

Figure 7-1 Configuration/Reconfiguration Flowchart

Shutdown Reconfiguration

Shutdown reconfiguration refers to the process of fully or partially reconfiguring the device without resetting, clearing all CRAMs, or exiting user mode. Shutdown reconfiguration requires a series of commands to be written in bitstream packet format to CCS to modify the configuration memory.

Table 7-13 Shutdown Reconfiguration via Slave SPI/Slave I^2C

Flow	Data
Write the "WREN" instruction	0x51
Write a CFG instruction to load:	0x50
100 padding bytes	0xFFFFFFFF …… 0xFFFFFFFF
SYNC WORD	0x01332D94
No operation	0xA0000000
Write an RSTCRC instruction to CMDR register	0xA8800001 0x00000001
Write to OPTION0R register (select "wakeup clock in shutdown mode")	0xAE400001 0x00000100/0x00000300(1)
Write to CMASKR register	0xADC00001
Write to CTRLR register (disable masked variable memory readback)	0x00000010
Write a SWAKEDOWN instruction to CMDR register	0xA8800001 0x00000008
20 No operation	0xA0000000 …… 0xA0000000
Write a GDOWN instruction to CMDR register	0xA8800001 0x0000000A
Write a DESYNC instruction to CMDR register	0xA8800001 0x0000000B
Bitstream	Bitstream
Write the "WRDIS" instruction	0x52
Write the "RDSR" instruction to read the device status register	0xA3

Note: (1) When the slave SPI interface is used, this value is 0x00000100, and when the slave I^2C interface is used, this value is 0x00000300.

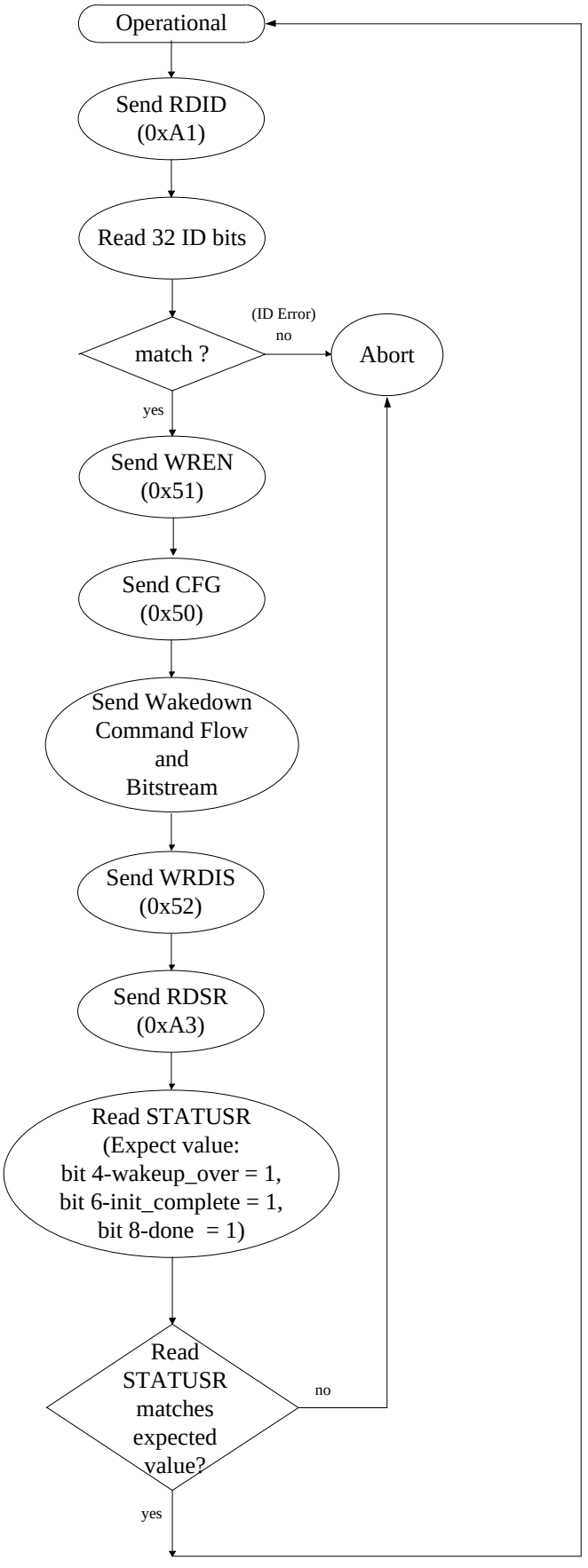

Figure 7-2 Shutdown and Reconfiguration Process

7.6 Programming embedded Flash via Slave SPI/Slave I^{2}C Interface

Embedded Flash supports direct operations through JTAG, slave SPI, and slave I^{2}C interfaces on the Embedded Flash. The operation process for slave SPI and slave I^{2}C are consistent, with differences only in interface timing. This section describes the operation process for slave SPI and slave I^{2}C. Refer to Chapter 8 Boundary Scan and JTAG Configuration for the JTAG operation process.

The complete process for programming embedded Flash via slave SPI and slave I^{2}C interfaces is as follows:

Figure 7-3 Programming embedded Flash 1

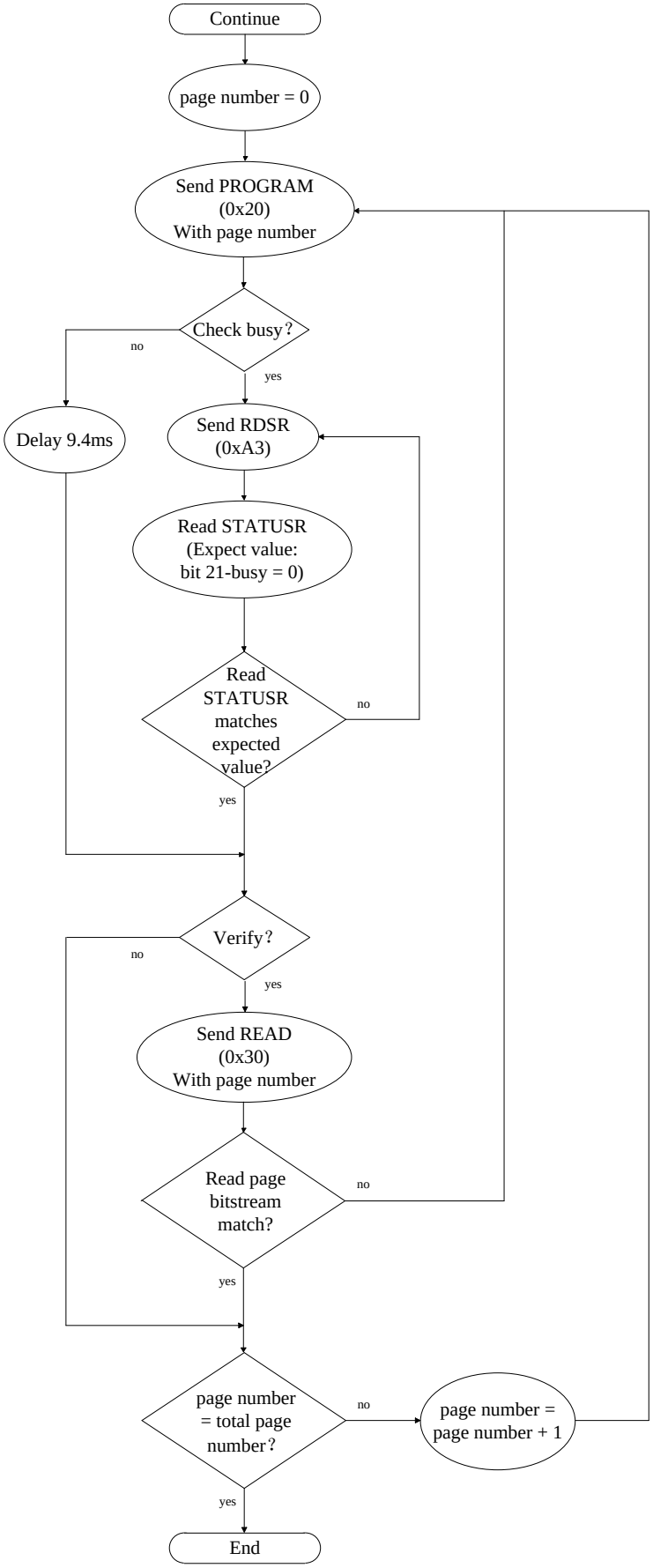

Figure 7-4 Programming embedded Flash 2

7.7 Programming embedded Flash via Internal Slave APB Interface

Compact family CPLDs support the operation of embedded Flash via an internal slave APB interface using user logic. The input to user logic can be via a JTAG interface or a user-defined interface. The block diagram of in-system indirect programming logic is shown below.

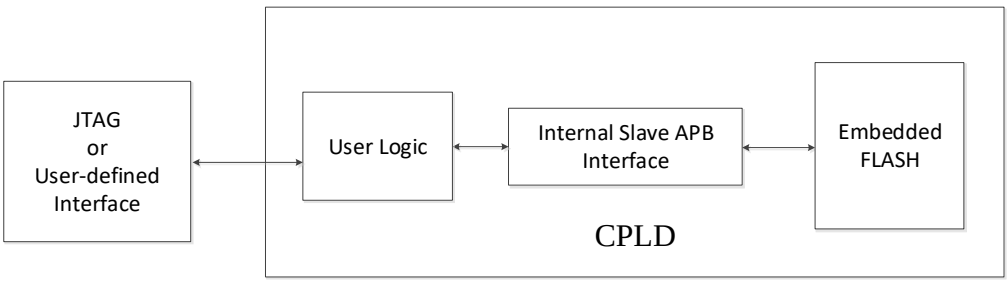

Figure 7-5 In-System Indirect Programming Logic Block Diagram

For the detailed slave APB operation process, refer to Appendix 7.

7.8 ISC Upgrade

PGC devices support ISC upgrade, which maintains the IO level status during the CRAM refreshing process and minimises the impact of IO state changes on the system during the upgrade.

PGC devices support ISC upgrades via JTAG, and SVF/PEF files with ISC upgrade processes can be generated using the Fabric Configuration tool. Additionally, PGC10KD also supports ISC upgrades via slave SPI and slave I^2C.

The following are important considerations during the ISC upgrade:

1. After an ISC upgrade, unless the device is repowered up, subsequent upgrades can only be performed using ISC instead of non-ISC.
2. ISC upgrades are generally performed in user mode. If done in configuration mode, the feature control bit CFG_DONE must be set as a general pin, and CFG_DONE must be pulled up by 4.7K to VCCIO0.
3. The ISC upgrade process requires the JTAG_EN pin to be pulled up by 4.7K to VCCIO0.

The JTAG ISC upgrade process is summarised in the figure below, where "IR" indicates the value of the "JTAG" instruction register, "LOG_WORK" indicates whether user logic is functioning, and "IO" indicates whether user IO is controlled by user logic or in a held state.

After the JTAG loads the "RESET" instruction, the device is reset and user logic stops working (LOG_WORK goes low). After other instructions are loaded, the reset is released, initiating initialization and Master Self Download. Once the Master Self Download is completed, the user logic starts functioning, but IO remains in a held state until the "IDCODE" instruction is loaded, then IO is released for user logic control.

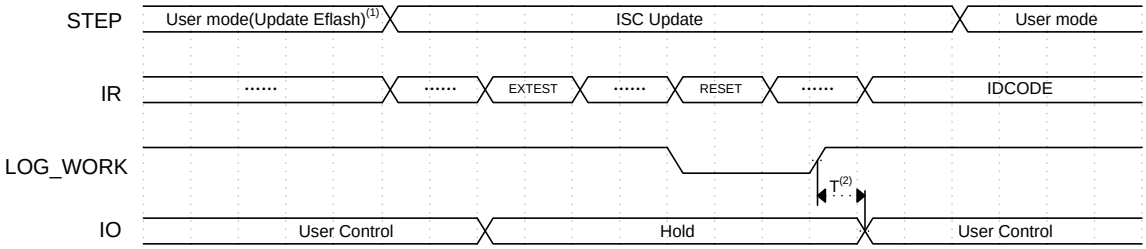

Figure 7-6 JTAG ISC Upgrade Process

Notes:

1. User mode (Update Eflash): Before an ISC upgrade, program the bitstream stored in Eflash in user mode. This process is by default in Background mode, meaning it does not affect the current logic function or IO status.
2. T: This time is by default the time taken to load three instructions plus 45ms (as defined in SVF/PEF), and then minus the device initialization and master self download time. Typically, the instruction loading time is far less than 45ms, however, as the instruction loading and the 45ms delay are generally executed by the CPU, the specific timings need to be determined based on actual circumstances.
3. When the ISC upgrade is initiated, the device will automatically sample and maintain the current IO state; it is required that before initiating the upgrade, the IO state is preserved through logical control or by pausing the user logic clock, to avoid erroneous sampling.

The ISC upgrade process via non-JTAG is summarised in the figure below, where "IR" indicates the value of the Slave SPI/ Slave I^2C instruction register, "LOG_WORK" indicates whether user logic is functioning, and "IO" indicates whether user IO is controlled by user logic or in a held state.

After the "RESET" instruction is loaded through slave SPI or slave I^2C, the device is reset, user logic stops working (LOG_WORK is pulled low), and initialization and Master Self Download are initiated, where the user IO is in held state. The host computer checks in a loop the status register by loading the "RDSR" instruction. When "wakeup_over" is 1, the "ISC_DISABLE" instruction is loaded to release IO for user logic control.

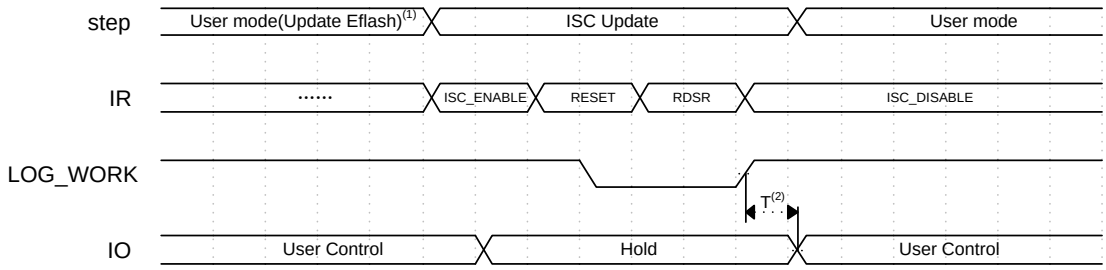

Figure 7-7 ISC Upgrade Process via non-JTAG

Notes:

1. User mode (Update Eflash): Before an ISC upgrade, program the bitstream stored in Eflash in user mode. This process is by default in Background mode, meaning it does not affect the current logic function or IO status.
2. T: This time is the duration for the host computer to check the status register and load the "ISC_DISABLE". The specific time is determined based on the actual situation.
3. When the ISC upgrade is initiated, the device will automatically sample and maintain the current IO state; it is required that before initiating the upgrade, the IO state is preserved through logical control or by pausing the user logic clock, to avoid erroneous sampling.

Chapter 8 Boundary Scan and JTAG Configuration

JTAG test refers to the application of test stimuli and analysis of test responses through JTAG pins to facilitate the fault diagnosis of the circuit under test.

The device supports IEEE1149.1 (Boundary Scan Test) and IEEE1532 (In-System Configuration of Programmable Devices) standards.

Boundary Scan Test (BST) refers to testing digital circuits using the JTAG bus and the chip's pin boundary scan cells.

8.1 System Block Diagram

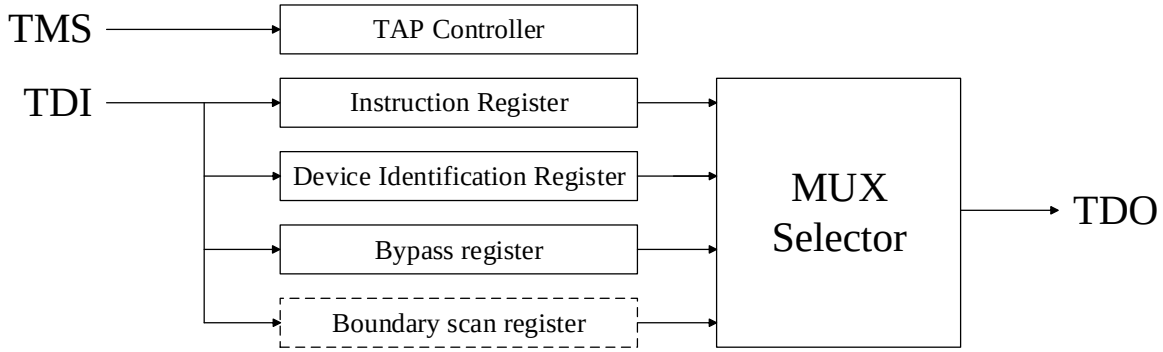

Figure 8-1 Block Diagram of JTAG Boundary Scan System

The hardware structure of the boundary scan consists of four parts:

- Test Access Port (TAP)
- Test Access Port Controller (TAPC)
- Instruction Register (IR)
- Test Data Register (TDR)

8.2 TAP Controller

The TAP controller is the core that controls the entire JTAG operation. The TAP controller, based on TCK and TMS signal changes, outputs various timings and modes required for the instruction register and test data register. It also generates control signals for the capture, shift, and update of the instruction register and test data register.

The TAP controller is a 16-state synchronous Finite State Machine (FSM). As shown in Figure 8-2, the state machine is divided into three columns corresponding to the reset operation, test data register operation and instruction register operation. The data on the status switching lines is

determined by the TMS signal level. The status switching of the TAP controller occurs on the rising edge of TCK.

For testing, the TAPC state machine consists of three basic actions: excitation (Update-DR state), execution (Test/Idle state), and response (Data Capture state). The actions for different types of tests are all subsets of these three basic actions. For example, an external test ("EXTEST" instruction) requires only two actions: stimulus and response.

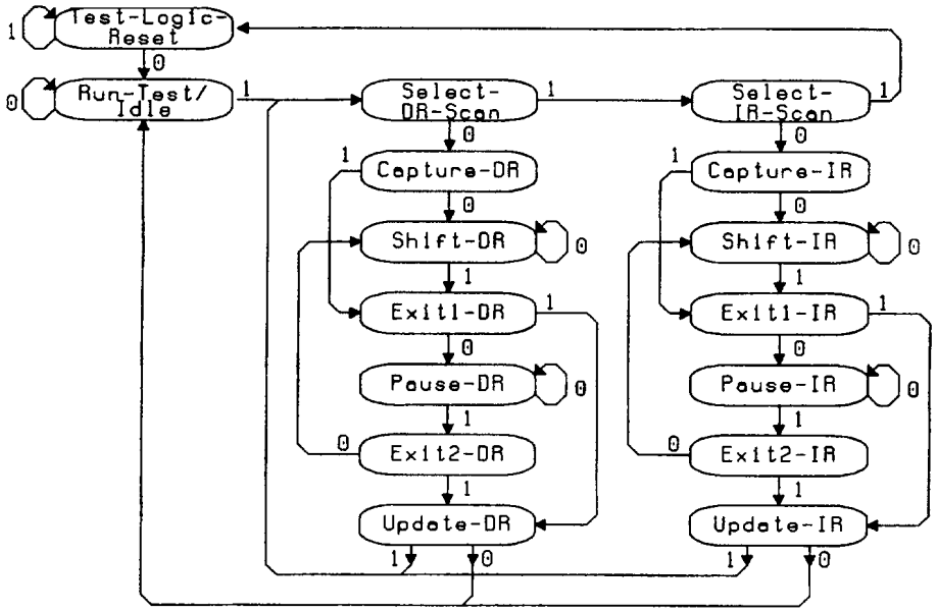

Figure 8-2 TAP Synchronous Finite State Machine (FSM)

Note: A detailed description of the states is shown in Appendix 8.

8.3 Instruction Register

The instruction register is 10 bits in length.

The instruction register receives and decodes instructions, generating control signals to select the test data register to be placed on the scan chain from TDI to TDO. It also controls the loading source and driving destination of the test data register.

When the TAPC enters the instruction capture state, the instruction register captures data "{1'b0, busy, wakedown_over, wakeup_over, init_n, init, isc_enabled, isc_done, 2'b01}" on the rising edge of TCK. "busy" indicates the embedded FLASH is in a busy state; "wakedown_over" indicates the completion of the wake-up shutdown process; "wakeup_over" indicates the completion of the wake-up process; "init_n" indicates the state of the INIT_FLAG_N signal; "init" indicates the completion of device initialization; "isc_enabled" indicates the enablement of JTAG ISC operation; and "isc_done" indicates the completion of JTAG ISC operation. The two lowest bits are fixed at 01 to detect the integrity of the scan chain and to locate faults.

When writing instructions to the instruction register, start with the lower bits.

When the TAPC enters the Update-DR state, the data from the shift register path is latched to the parallel output of the instruction register on the falling edge of TCK. Once the new instruction is latched, it becomes the current instruction.

8.4 Instruction Set

Table 8-1 JTAG Instruction Set

Instruction	Types	Op Code	Description
BYPASS	1149.1 Non-test instruction	1111111111	Bypass instruction
SAMPLE/PRELOAD	1149.1 Non-test instruction	1010000000	Sample/preload instruction
EXTEST	1149.1 Test instruction	1010000001	External test instruction
INTEST	1149.1 Test instruction	1010000010	Internal test instruction
IDCODE	1149.1 Non-test instruction	1010000011	Identification instruction
HIGHZ	1149.1 Test instruction	1010000101	High-Z instruction
JRST	Only for design	1010001010	Reset instruction
CFGI	Only for design	1010001011	Configuration instruction
CFGO	Only for design	1010001100	Readback instruction
JWAKEUP	Only for design	1010001101	Wakeup instruction
READ_UID	Only for design	0101001100	Read-UID instruction
RDSR	Only for design	0101011001	Read-Status-Register instruction
WADR	Only for design	0101011010	Write-Address instruction
ERASE	Only for design	0101011101	Erase instruction
ERASE_PAGE	Only for design	0101011110	Page-Erase instruction
ERASE_CTL	Only for design	0101011111	Control-Erase instruction
PROGRAM	Only for design	0101100000	Program instruction
PROGRAM_CTL	Only for design	0101100001	Program-Control instruction
READ	Only for design	0101100010	Read instruction
READ_CTL	Only for design	0101100011	Control-Read instruction
PROGRAM_LOCK	Only for design	0101100100	Lock-Embedded-Flash instruction
READ_LOCK	Only for design	0101100101	READ-Embedded-Flash-Lock-Flag instruction
EFlash_SLEEP	Only for design	0101100110	Embedded-Flash-Sleep instruction
EFlash_WAKEUP	Only for design	0101100111	Embedded-Flash-Wakeup instruction

Note: For a detailed description of "JTAG" instructions, refer to Appendix 1.

8.5 Test Data Register (TDR)

Table 8-2 List of Test Data Registers

Property	Item	Bit width
Data register	Device identification register	32
	Bypass register	1
	Boundary scan register	2
	UID register	64
	Feature control register	32
	Program register	32
	Status register	32
	Address register	24
	LOCK register	12
Internal data register	READ register	-
	Configuration register	-

Table 8-3 Instruction and Test Data Register Mapping Table

Instruction	Data register
BYPASS	Bypass register
SAMPLE/PRELOAD	Boundary scan register
EXTEST	Boundary scan register
INTEST	Boundary scan register
IDCODE	Device identification register
USERCODE	Device identification register
HIGHZ	Bypass register
JRST	Bypass register
CFGI	Bypass register
CFGO	Configuration register
JWAKEUP	Bypass register
READ_UID	UID register
RDSR	Status register
WADR	Address register
ERASE	Bypass register
ERASE_PAGE	Bypass register
ERASE_CTL	Bypass register
PROGRAM	Program register
PROGRAM_CTL	Feature control register
READ	READ register
READ_CTL	Feature control register
PROGRAM_LOCK	LOCK register
READ_LOCK	LOCK register

Instruction	Data register
EFlash_SLEEP	Bypass register
EFlash_WAKEUP	Bypass register

8.5.1 Device identification register

The device identification register is 32 bits in length.

The device identification register is capable of capturing and shifting data without parallel output.

When the current instruction is IDCODE and the TAPC state machine is in the data capture state, the device identification register captures the device flags in parallel on the rising edge of TCK. The PGC family device IDs can be found in the Device ID Register (IDR).

8.5.2 Bypass register

When certain chips need isolation, a bypass register can be used to achieve short-circuiting and shorten the length of the entire boundary scan chain.

The bypass register is 1 bit in length.

The bypass register can perform capture and shift operations without parallel output.

When the TAPC state machine is in the data capture state, the bypass register captures a logical 0 on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the bypass register captures TDI on the rising edge of TCK.

The bypass register does not have parallel output.

8.5.3 Boundary scan register

The boundary scan register is a collection of boundary scan units. Boundary scan units are placed at the input port, output port, bidirectional port, and tri-state port of the device signal. Combining the boundary scan units forms a boundary scan register.

Pin connections and placement constraints determine the connection order of the boundary scan units.

The multiplexed TCK, TMS, TDI, and TDO of PADs have no boundary scan register.

All other PADs have the same boundary scan register as the bi-directional DC PAD.

The boundary scan register of bi-directional DC PAD uses BC_2 control + BC_7 data

8.5.4 UID register

The UID register is used for reading the UID.

The UID register is 64 bits in length.

The UID register is capable of capturing, shifting, and updating data.

The current instruction is READ_UID:

When the TAPC state machine is in the data capture state, the UID register captures the UID stored in the embedded FLASH on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the UID register captures the value of TDI on the rising edge of TCK.

8.5.5 Feature control register

The feature control register is used for programming and reading feature control bits from the embedded Flash.

The feature control register is capable of capturing, shifting, and updating data.

The current instruction is PROGRAM_CTL:

When the TAPC state machine is in the data capture state, the feature control register captures the feature control bits stored in the embedded FLASH on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the feature control register captures the value of TDI on the rising edge of TCK.

When the TAPC state machine is in the data update state, the data from the shift register path is latched to the parallel output of the feature control register on the falling edge of TCK.

The current instruction is READ_CTL:

When the TAPC state machine is in the data capture state, the feature control register captures the feature control bits stored in the embedded FLASH on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the feature control register captures the value of TDI on the rising edge of TCK.

8.5.6 Program register

The PROGRAM register is used for programming the embedded FLASH.

The PROGRAM register is 32 bits in length.

The PROGRAM register is capable of shifting and updating data.

The current instruction is PROGRAM:

When the TAPC state machine is in the data shift state, the PROGRAM register captures the value of TDI on the rising edge of TCK.

When the TAPC state machine is in the data update state, the data from the shift register path is latched to the parallel output of the PROGRAM register on the falling edge of TCK.

8.5.7 Status register

The status register is used for reading the status register of the configuration control system.

The status register is 32 bits in length.

The status register is capable of capturing and shifting data, without parallel output.

The current instruction is RDSR:

When the TAPC state machine is in the data capture state, the status register captures the status register of the configuration control system on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the status register shifts right by one bit on the rising edge of TCK.

8.5.8 Address register

The address register is used for reading and writing the 24-bit starting address of the embedded FLASH.

The address register is 24 bits in length.

The address register is capable of capturing, shifting, and updating data.

The current instruction is WADR:

When the TAPC state machine is in the data capture state, the address register captures the current starting address of the embedded FLASH on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the address register shifts right by one bit on the rising edge of TCK.

When the TAPC state machine is in the data update state, the data from the shift register path is latched to the parallel output of the address register on the falling edge of TCK.

8.5.8.1 1K/2K/4K/7K

Table 8-4 Address Register (1)

Address	**Item**	**Description**
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address

Address	Item	Description
[16:8]	Ra[8:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2’b11.

8.5.8.2 10K

Table 8-5 Address Register (2)

Address	Item	Description
[23:20]	Reserved	Must be set to 5'b00000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[8:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

8.5.9 LOCK register

The LOCK register is used for programming and reading the embedded Flash lock flags and the end page address of the Master Self Download bitstream.

The 1K/2K/4K/7K LOCK register is 12 bits in length (lock flag: lock; heap address: lock_ba[1:0]; row address: lock_ra[8:0]).

The 10K LOCK register is 13 bits in length (lock flag: lock; heap address: lock_ba[1:0]; row address lock_ra[9:0]).

For 1K devices, lock_ba[1:0] is the reserved bit.

For 2K devices, lock_ba[1:0] is the reserved heap address and should be set to 2'b11.

The LOCK register is capable of capturing, shifting, and updating data.

The current instruction is PROGRAM_LOCK:

When the TAPC state machine is in the Data Capture state, the LOCK register captures the embedded Flash lock flag and the end page address of the Master Self Download bitstream on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the LOCK register shifts right by one bit on the rising edge of TCK.

When the TAPC state machine is in the data update state, the data from the shift register path is latched to the parallel output of the LOCK register on the falling edge of TCK.

The current instruction is READ_LOCK:

When the TAPC state machine is in the Data Capture state, the LOCK register captures the embedded Flash lock flag and the end page address of the Master Self Download bitstream on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the LOCK register shifts right by one bit on the rising edge of TCK.

8.5.10 READ register

The READ register is used to read the contents of embedded FLASH.

The READ register is an internal data register that is not placed on the TDI to TDO scan chain.

The current instruction is READ. When the TAPC state machine is in the data shift state, the PROGRAM register shifts left by one bit on the rising edge of TCK.

8.5.11 Configuration register

The JTAG configuration register is used to read back the configuration register and CCS configuration register.

The configuration register is an internal data register that is not placed on the TDI to TDO scan chain.

The current instruction is CFGO. When the TAPC state machine is in the data shift state, the configuration register shifts left by one bit on the rising edge of TCK.

8.6 Operation Flow

8.6.1 CRAM Configuration

Configuration/Reconfiguration

Configuration refers to the process of configuring a CPLD chip from the start until it enters user mode; Reconfiguration refers to the process of resetting (JTAG soft reset or externally triggered INIT_FLAG_N hard reset), clearing all CRAM, exiting user mode, and then reconfiguring after the CPLD chip enters user mode. The flow is as follows:

Table 8-6 Configuration/Reconfiguration Process

Steps	Operation Description
1	Load the "JRST" instruction into IR, starting with the least significant bit
2	Enter the Run-Test/Idle state and keep at least 1 TCK cycle
3	Cycle into the CAPTURE-IR state and load the "CFGI" instruction into IR until init_n is detected high, starting with the least significant bit
4	Enter Shift-DR state and load the bitstream, starting with the high bits
5	Enter the Test Logic Reset state
6	Load the "JWAKEUP" instruction into IR, starting with the least significant bit
7	Enter the Run-Test/Idle state, to last for a minimum of 30 TCK cycles
8	(User clock wake-up, to last for a minimum of 30 UCLK cycles)
9	Load the "CFGI" instruction into IR, starting with the least significant bit
10	Enter the Run-Test/Idle state, to last for a minimum of 125 TCK cycles
11	Enter Shift-DR state:
12	Write a pad word (FFFFFFFF), starting with the high bits
13	Write the synchronization word (01332D94), starting with the high bits
14	Write a NOP Packet Header (00000000), starting with the high bits
15	Write a header with a STATUSR register (B2400001), starting with the high bits
16	Load the "CFGO" instruction into IR, starting with the least significant bit
17	Enter Shift-DR state, shifting out the contents of the STATUSR register, starting with the most significant bit
18	Enter the Test Logic Reset state

8.6.2 embedded Flash Programming

After device initialization is completed, it needs to wait 100μs before proceeding with operations.

For the detailed JTAG operation process, refer to Appendix 2, where the number of continuous clock cycles in the Run Test Idle state is determined based on a fixed delay time with TCK at 50MHz; users can determine the number of clocks based on the actual TCK frequency. Users can determine the number of clocks to wait based on the actual TCK frequency. If TCK is at 50MHz, 100 cycles are needed; if at 5MHz, 10 clock cycles are required.

8.6.3 One-Step Test

One-step test controls the input ports through the boundary scan register, applying stimuli to the chip's input ports, capturing the chip's output port response data into the boundary scan register, and serially shifting the data out through the TDO port.

Table 8-7 One-Step Test

Steps	**Operation Description**
1	Load the "PRELOAD" instruction
2	Send the first test vector (input stimulus serial shift-in data and parallel update to the boundary scan register)
3	Load the "INTEST" instruction (input control changes from being input port-driven to boundary scan register-driven)
4	While sending the second test vector, the first response data is shifted out from the TDO port for response analysis and fault diagnosis (input stimulus serial shift-in while output response serial shift-out, and then the input stimulus is parallelly updated in the boundary scan register)
5	While sending the third test vector, the second response data is shifted out from the TDO port for response analysis and fault diagnosis (input stimulus serial shift-in while output response serial shift-out, and then the input stimulus is parallelly updated in the boundary scan register)
6	……
7	While sending the (n-1)th test vector, the nth response data is shifted out from the TDO port for response analysis and fault diagnosis (input stimulus serial shift-in while output response serial shift-out, and then the input stimulus is parallelly updated in the boundary scan register)

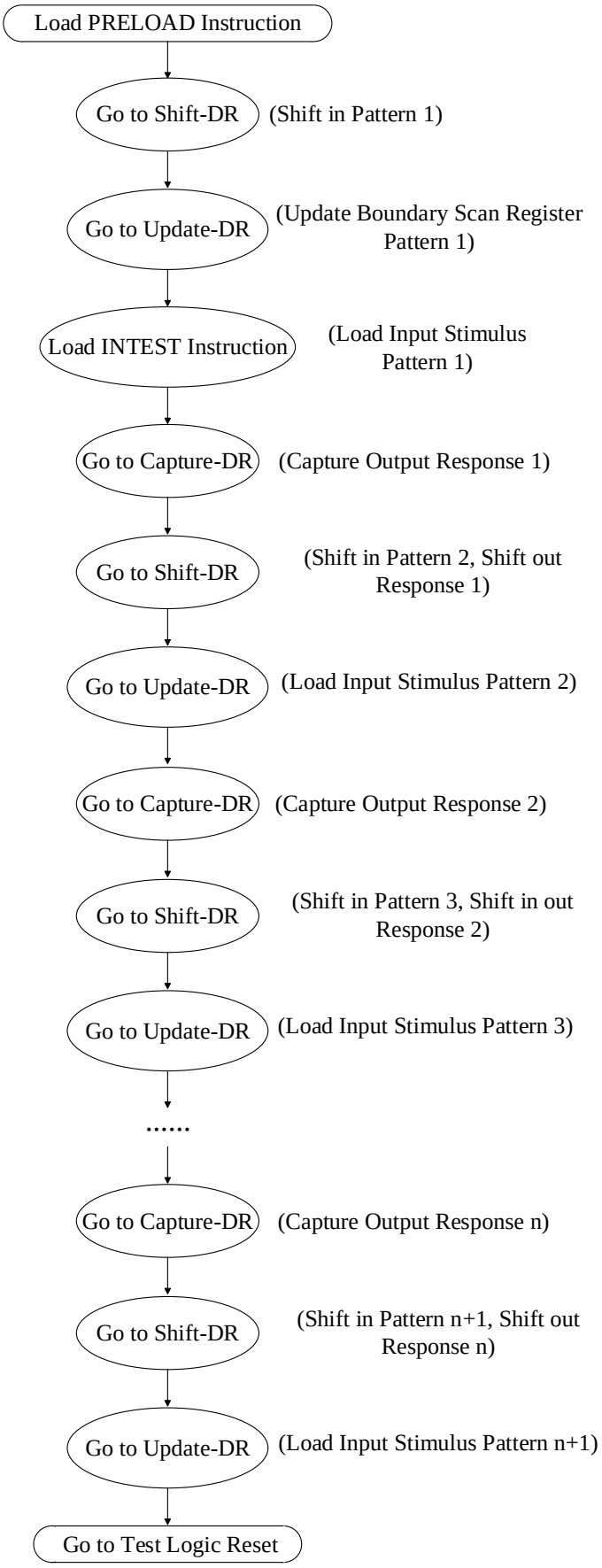

Figure 8-3 Boundary Scan One-Step Test

8.6.4 Interconnect Test

The interconnect test generates chip output port stimuli, controlled via the boundary scan register. Response data from the chip input ports are captured into the boundary scan register and are serially shifted out through the TDO port.

Table 8-8 Interconnect Test

Load the "PRELOAD" instruction
Send the first test vector (input stimulus is serially shifted in and parallelly updated to the boundary scan register)
Load external test instruction (EXTEST) (output control changes from being output port-driven to boundary scan register-driven)
While sending the second test vector, the first response data is shifted out from the TDO port for response analysis and fault diagnosis (output stimulus serial shift-in while input response serial shift-out, and then the output stimulus is parallelly updated in the boundary scan register)
While sending the third test vector, the second response data is shifted out from the TDO port for response analysis and fault diagnosis (output stimulus is serially shifted in while input response is serially shifted out, and then the output stimulus is parallelly updated in the boundary scan register)
……
While sending the (n-1)th test vector, the nth response data is shifted out from the TDO port for response analysis and fault diagnosis (output stimulus serial shift-in while the input response serial shift-out, and then the output stimulus is parallelly updated in the boundary scan register)

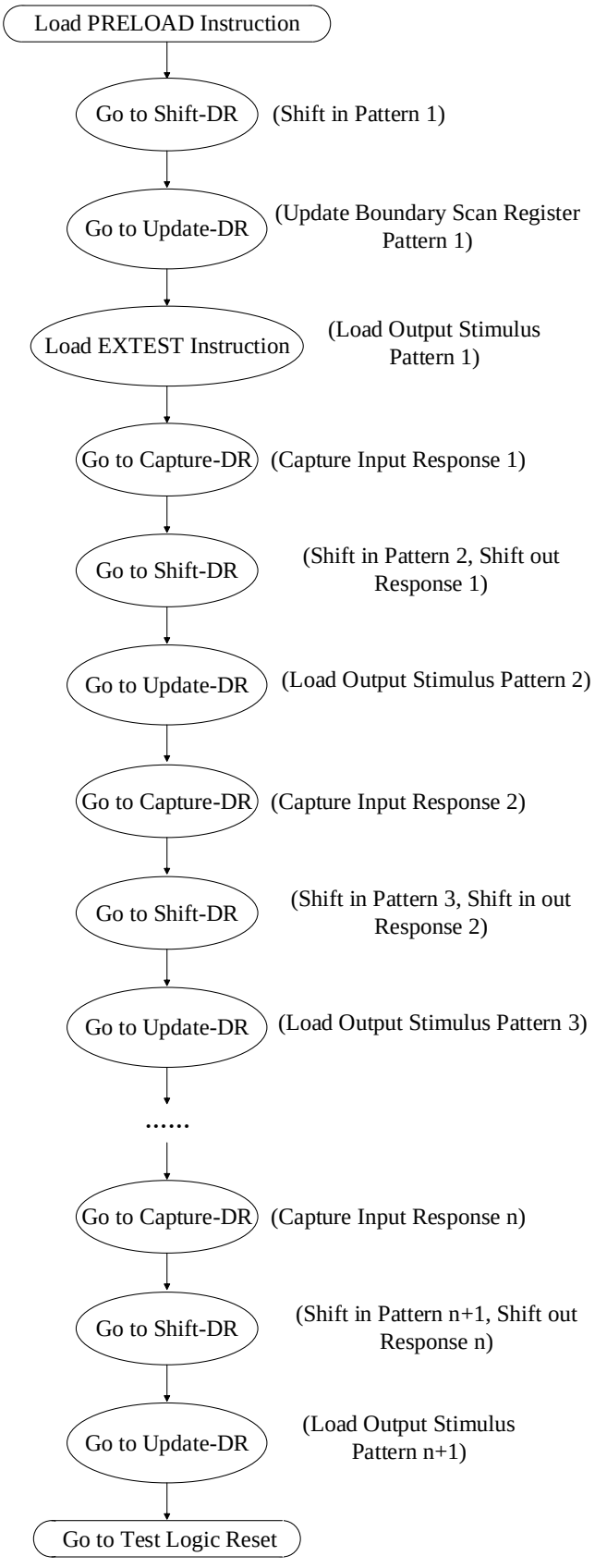

Figure 8-4 Boundary Scan Interconnect Test

Chapter 9 User Logic Interfaces

CPLD devices support six types of user logic interfaces: JTAG user instruction interface, UID interface, power controller, user wake-up interface, internal slave APB interface, and readback CRC. For detailed GTP usage instructions, refer to the PDS software installation appendix as shown in "... \PDS_xxx\arch\vendor\pango\verilog\simulation".

9.1 JTAG User Instruction Interface (SCANCHAIN)

TCK, TMS, and TDI are inputted from PAD to Fabric.

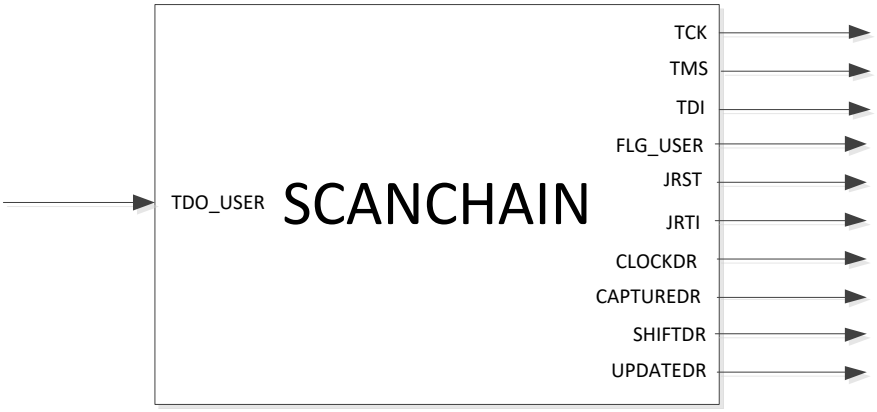

Figure 9-1 JTAG User Instruction Interface

9.1.1 Port List

Table 9-1 JTAG User Port List

Item	I/O	Description
TCK	O	JTAG clock, with a maximum frequency of 50MHz
TMS	O	JTAG test mode select. Control the state transition of the TAP controller on the rising edge of the clock TCK
TDI	O	JTAG Test Data Input. Serve as the serial input for JTAG instructions and data registers on the rising edge of the clock TCK
FLG_USER	O	User-defined register instruction flag
TDO_USER	I	User-defined register data input
JRST	O	User JTAG soft reset
JRTI	O	User JTAG test/idle
CLOCKDR	O	Gated clock, used for user-defined register instruction operations
CAPTUREDR	O	User CAPTURE-DR flag
SHIFTDR	O	User Shift-DR flag
UPDATEDR	O	User UPDATE-DR flag

9.1.2 Interface timing

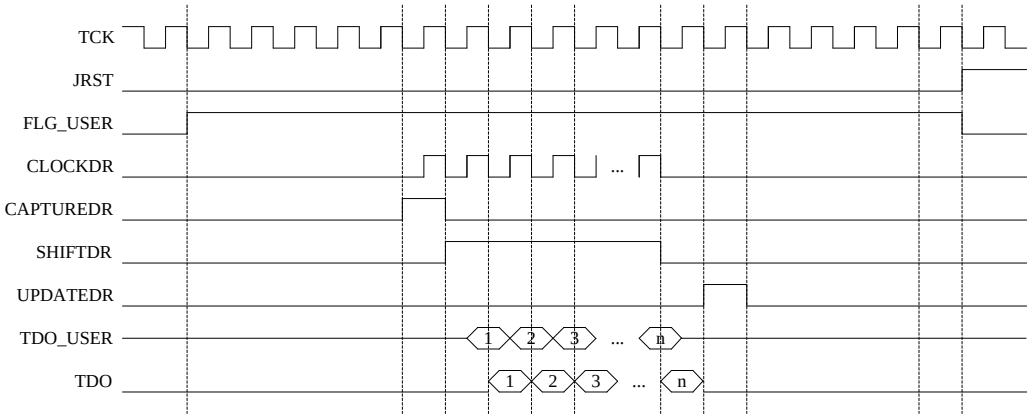

Figure 9-2 JTAG User Interface Timing Diagram

9.2 UID Interface (UDID)

The 64-bit UID is stored in embedded Flash and is programmed uniformly when the device is manufactured. Each time the device is powered up, the UID from the embedded Flash is automatically read into the register for user access at any time. Of the 64 bits, the higher 8 bits are a user-defined field, and the lower 56 bits are for factory use.

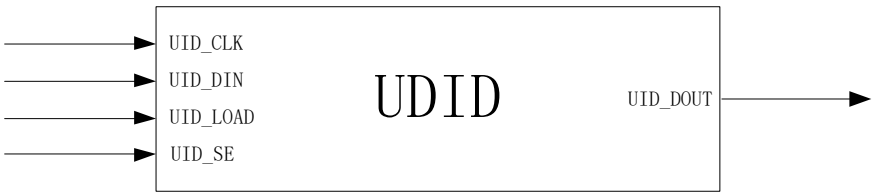

Figure 9-3 UID Interface Diagram

9.2.1 Port List

Table 9-2 UID Port List

Item	I/O	Description
UID_CLK	I	UID Clock
UID_DIN	I	UID Serial Data Input
UID_LOAD	I	UID Parallel Data Load
UID_SE	I	UID Serial Data Shift
UID_DOUT	O	UID Serial Data Output

9.2.2 Interface timing

- The UID is 64-bit long.

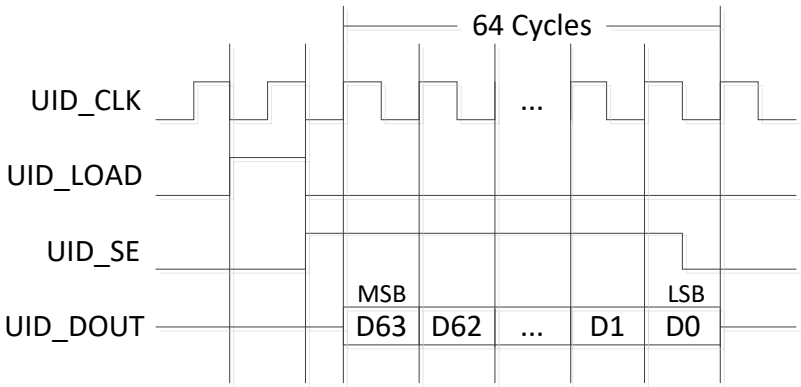

Figure 9-4 64-Bit UID Interface Timing Diagram

Users can extend the bit width of the UID.

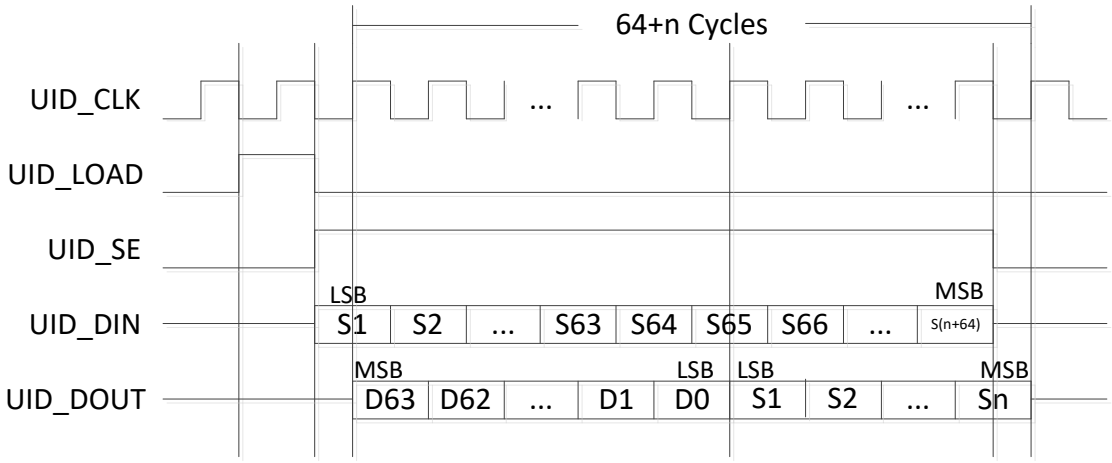

Figure 9-5 Extended UID Timing Diagram

9.3 Power Controller

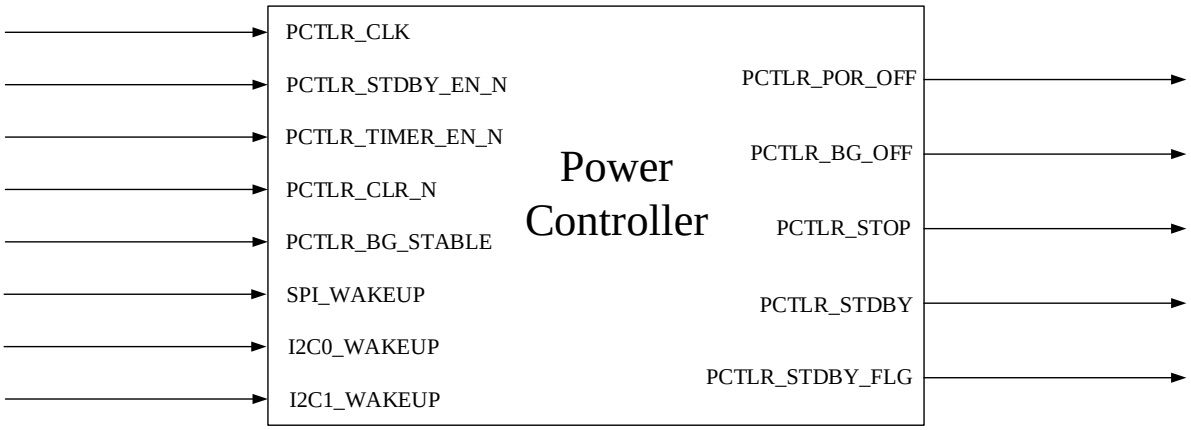

Figure 9-6 Power Controller Interface Diagram

9.3.1 Port List

Table 9-3 Power Controller Port List

Item	I/O	Source/Purpose	Description
PCTLR_CLK	I	SRB	Clock, with a maximum frequency of 50M
PCTLR_STDBY_EN_N	I	SRB	Standby enable, active low Enter the shutdown process from operating mode to standby mode, starting from the falling edge of standby enable. Enter the wake-up process from standby mode to operating mode, starting from the rising edge of standby enable
PCTLR_TIMER_EN_N	I	SRB	Timer enable, active low Pulse signal, for starting the timer counting
PCTLR_CLR_N	I	SRB	Standby complete, active low Pulse signal, for clearing the standby flag after wake-up
PCTLR_BG_STABLE	I	Band gap circuit	Stable Band Gap Circuit
PCTLR_POR_OFF	O	Power-up Reset Circuit	Power-Up Reset Circuit Shutdown
PCTLR_BG_OFF	O	Band gap circuit	Band gap circuit shutdown When the band gap circuit is shut down, the power-up reset circuit and analogue circuits (PLL, OSC, differential IO) are turned off
PCTLR_STOP	O	SRB	Stop Used for standby preparation. User logic prepares for standby by shutting down signals such as the clock
PCTLR_STDBY	O	SRB	Standby Global signal, output via SRB to user logic, IO PAD, and PLL. Used for controlling logic circuits to enter standby mode
PCTLR_STDBY_FLG	O	SRB	Standby flag Indicates that the device is in standby mode
SPI_WAKEUP	I	SPI functional module	SPI functional module wake-up request When the wake-up enable bit of the SPI functional module control register is active, if the SPI functional module is selected as a slave device (chip select input is valid), then the SPI functional module sends a wake-up request to the power controller
I2C0_WAKEUP	I	I^2C0 functional modules	I^2C0 functional module wake-up request When the wake-up enable bit of the I^2C0 functional module control register is active, if the I^2C0 functional module is selected as a slave device (slave device address match), then the I^2C0 functional module sends a wake-up request to the power controller
I2C1_WAKEUP	I	I^2C1 functional module	I^2C1 functional module wake-up request When the wake-up enable bit of the I^2C1 functional module control register is active, if the I^2C1 functional module is selected as a slave device (slave device address match), then the I^2C1 functional module sends a wake-up request to the power controller

9.3.2 Mode Conversion

The power controller has four modes: shutdown mode, standby mode, wake-up mode, and operating mode. The conversion of modes is controlled by the state machine constituted by PCTLR_STOP and PCTLR_STDBY signals. The mode conversion and truth table are as follows.

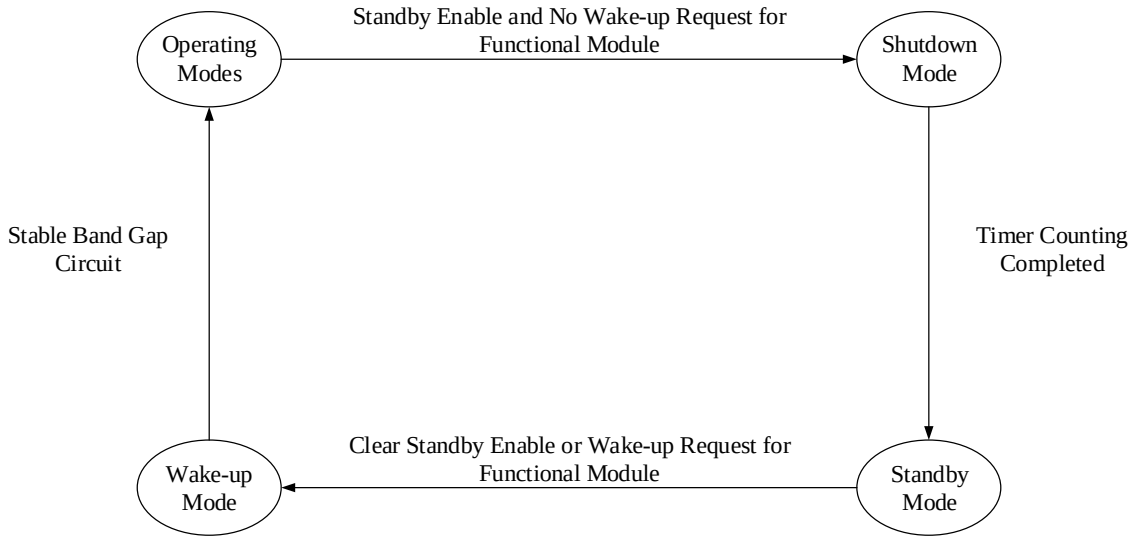

Figure 9-7 Power Controller Mode Conversion Diagram

Table 9-4 Power Controller Mode Truth Table

	PCTLR_STOP	**PCTLR_STDBY**
Operating Modes	0	0
Shutdown Mode	1	0
Standby Mode	1	1
Wake-Up Mode	0	1

9.3.3 Interface timing

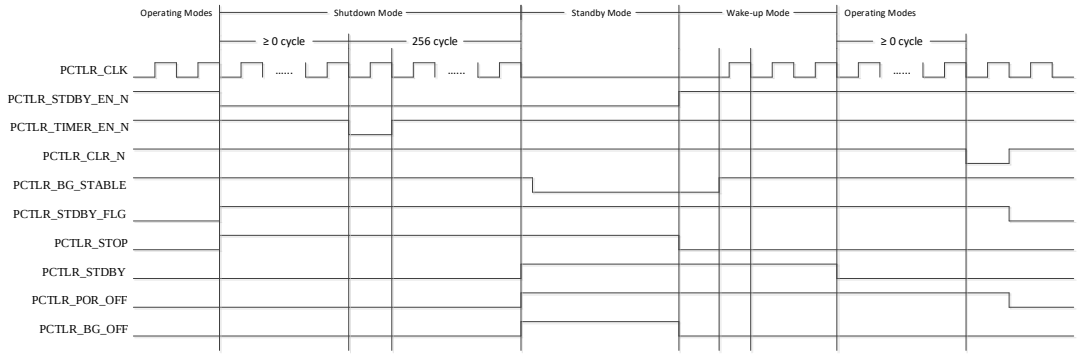

Figure 9-8 Power Controller Interface Timing Diagram

Note that a stable clock is only available after the band gap circuit stabilises.

9.3.3.1 Standby

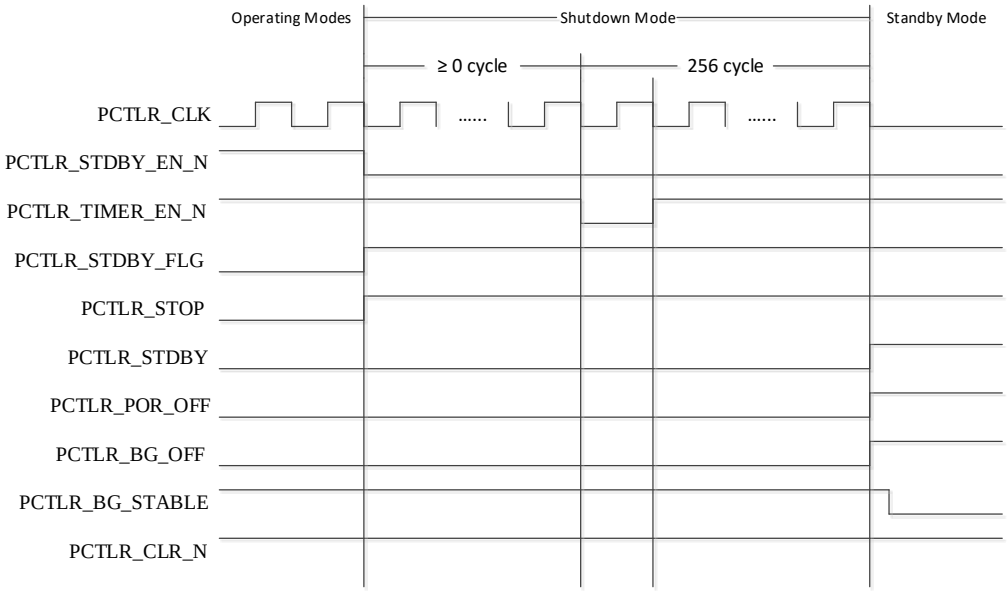

Figure 9-9 Standby Timing Diagram

1. When the standby enable STDBY_EN_N changes to active from inactive, the power controller enters shutdown mode from operating mode:
   1) Generates standby flag STDBY_FLG, indicating the device is in standby mode.
   2) Generates a standby preparation STOP, where user logic prepares for standby through signals like shutdown clock.
2. The timer enable TIMER_EN_N pulse becomes active, starting the timer counting.
3. After counting 256 clock cycles by the timer:
   1) Generates a global standby signal which is outputted via SRB to user logic, IO PAD, OSC, and PLL for controlling logic circuits to enter standby mode.
   2) Shuts down the band gap circuit and the power-up reset circuit. When the band gap circuit is shut down, the power-up reset circuit and analogue circuits (PLL, OSC, differential IO) are also shut down.

9.3.3.2 Wakeup

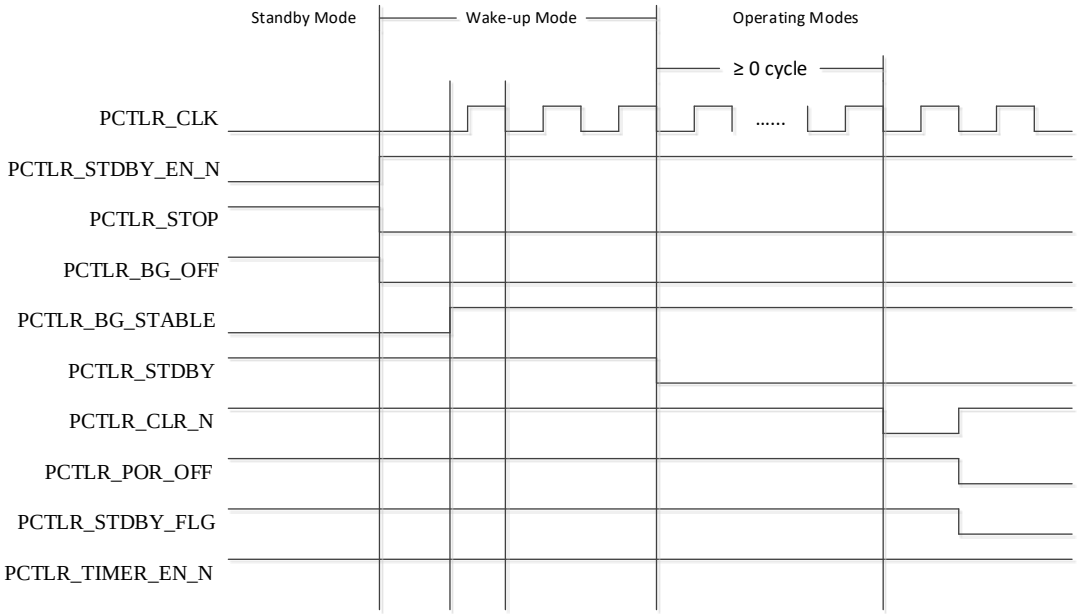

Figure 9-10 Wake Timing Diagram

1. When the standby enable STDBY_EN_N changes from active to inactive, the power controller enters wake-up mode from standby mode:
   1) Clears standby preparation STOP.
   2) Turns on the band gap circuit.
2. After the band gap circuit stabilises, the global standby signal is cleared, and the power controller enters operating mode from wake-up mode.
3. After entering the operating mode, the user logic generates a standby end pulse:
   1) Turns on the power-up reset circuit.
   2) Clears the standby flag STDBY_FLG.

9.3.3.3 SPI/I^{2}C Functional Module Wake-Up

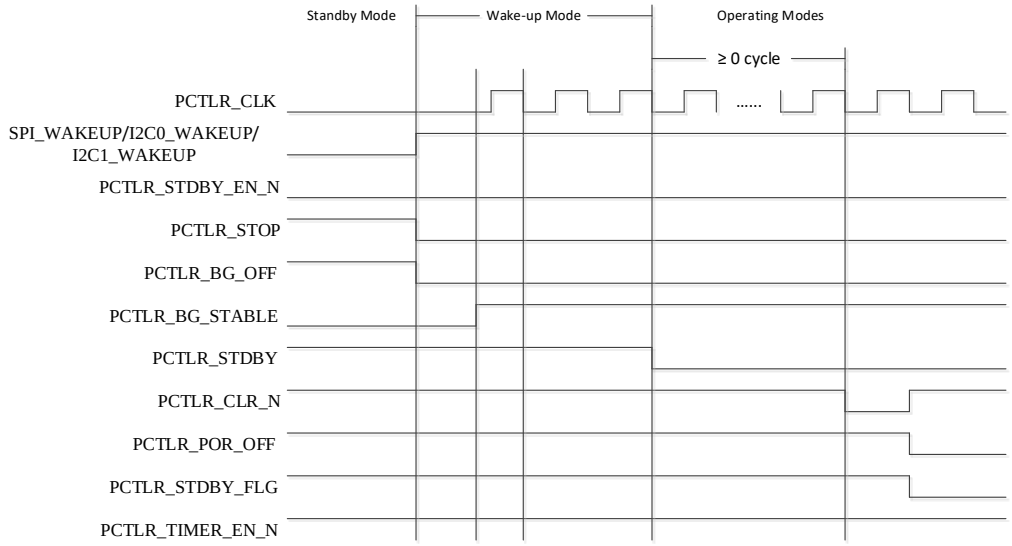

Figure 9-11 SPI/I^{2}C Functional Module Wake-up Timing Diagram

When the wake-up enable bit of the SPI functional module control register is active, if the SPI functional module is selected as a slave device (chip select input is valid), then the SPI functional module sends a wake-up request to the power controller.

When the wake-up enable bit of the I^{2}C functional module control register is active, if the I^{2}C functional module is selected as a slave device (slave device address matches), then the I^{2}C functional module sends a wake-up request to the power controller.

1. When a functional module sends a wake-up request (SPI_WAKEUP or I2C0_WAKEUP or I2C1_WAKEUP), the power controller transitions from standby mode to wake-up mode:
    1) Clears standby preparation STOP.
    2) Turns on the band gap circuit.
2. After the band gap circuit stabilises, the global standby signal is cleared, and the power controller enters operating mode from wake-up mode.
3. After entering the operating mode, the user logic generates a standby end pulse:
    1) Turns on the power-up reset circuit.
    2) Clears the standby flag STDBY_FLG.

9.3.3.4 SPI/I^{2}C Functional Module Standby

After the SPI/I^{2}C functional module finishes the mode operation, it cancels the wake-up request. At this point, if the user standby enable is active and no other functional modules have a wake-up request, then the power controller returns to standby mode.

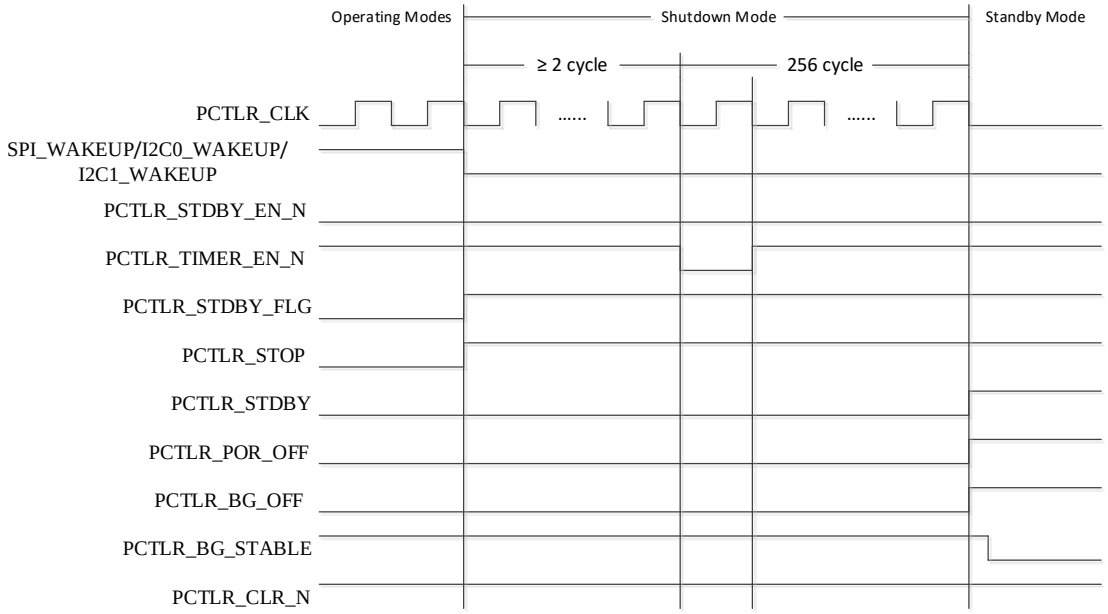

Figure 9-12 SPI/I^{2}C Functional Module Standby Timing Diagram

1. When the standby enable STDBY_EN_N changes to active from inactive, the power controller enters shutdown mode from operating mode:
   1) Generates a standby flag STDBY_FLG, indicating that the device is in standby mode.
   2) Generates a standby preparation STOP, where user logic prepares for standby through signals like shutdown clock.
2. The timer enable TIMER_EN_N pulse becomes active, starting the timer counting.
3. After counting 256 clock cycles by the timer:
   1) Generates a global standby signal which is outputted via SRB to user logic, IO PAD, OSC, and PLL for controlling logic circuits to enter standby mode.
   2) Shuts down the band gap circuit and the power-up reset circuit. When the band gap circuit is shut down, the power-up reset circuit and analogue circuits (PLL, OSC, differential IO) are also shut down.

9.4 User Wake-up Interface (START)

Figure 9-13 User Wake-up Interface Diagram

9.4.1 Port List

Table 9-5 User Wake-up Port List

Item	I/O	Description
UCLK	I	User Clock
GOUTEN_I	I	User Output Enable
GRSN_I	I	User Global Reset
GWEN_I	I	User Memory Write Enable

9.4.2 Parameter

Table 9-6 User Wake-up Parameters

Item	Defaults	Description
CP_GOUTEN_EN	DISABLE	User Output Enable Effective ENABLE: Enabled DISABLE: Disabled
CP_GRSN_EN	DISABLE	User Global Reset Effective ENABLE: Enabled DISABLE: Disabled
CP_GWEN_EN	DISABLE	User Memory Write Enable Effective ENABLE: Enabled DISABLE: Disabled

9.5 Internal Slave APB Interface

The internal slave APB interface of the CCS is an APB slave device interface, compatible with AMBA APB3 protocol and supporting 7 slave devices: CCS, SPI, I^2C0, I^2C1, TIMER, PLL0, and PLL1.

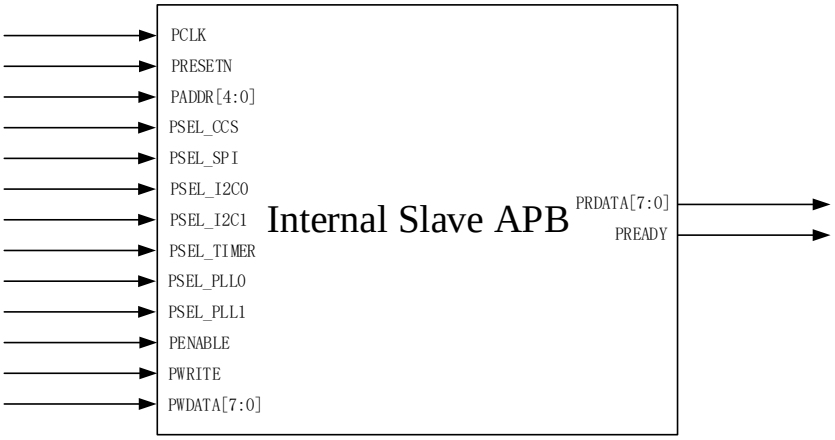

Figure 9-14 Internal Slave APB Interface Diagram

9.5.1 Port List

Table 9-7 Internal Slave APB Port List

Item	I/O	Description
pclk	I	Clock, sampled on the rising edge
presetn	I	Asynchronous reset, active low
paddr[4:0]	I	Address Bus
psel	I	Indicates that at the current moment, only one of the seven slave devices can be selected
penable	I	Enabled Indicates the second and subsequent cycles of transmission.
pwrite	I	Orientation 0: Read 1: Write
pwdata[7:0]	I	Data bus input
prdata[7:0]	O	Data bus output
pready	O	Ready Indicates the end of a normal bus cycle

9.5.2 Interface timing

Write

Write without wait

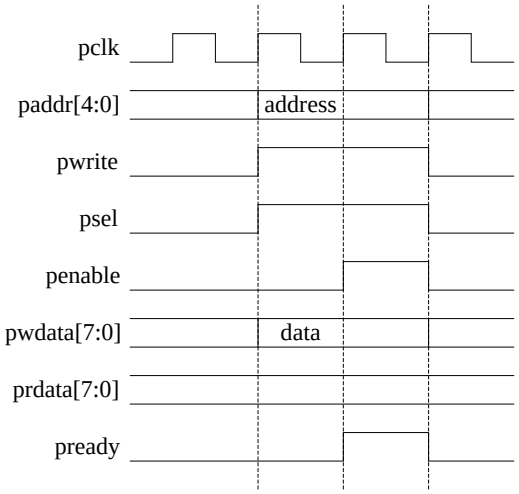

Figure 9-15 Write Without Wait

Write with wait

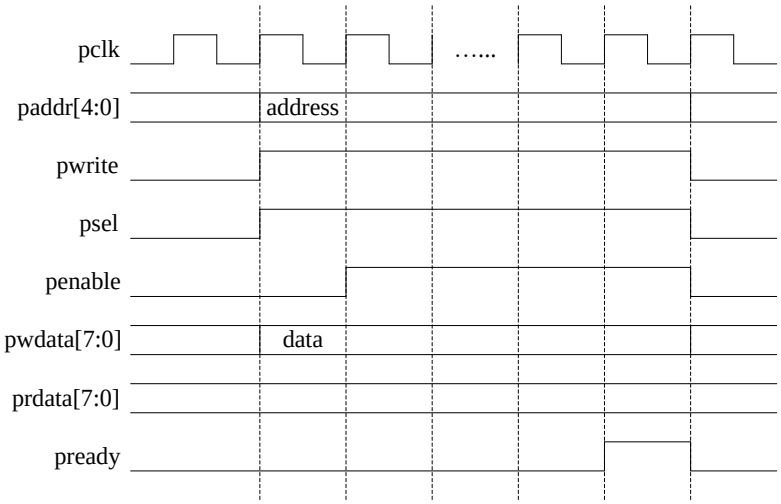

Figure 9-16 Write With Wait

Continuous Write

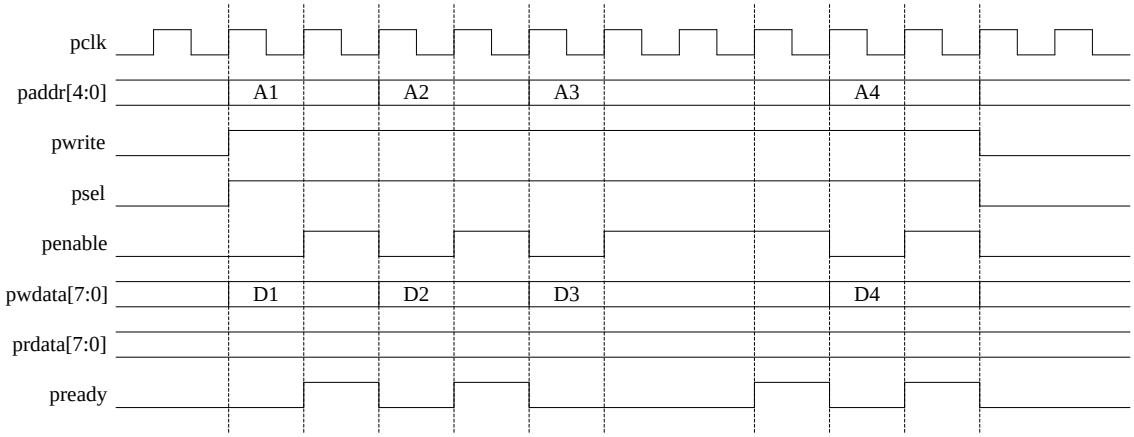

Figure 9-17 Continuous Write

Read

Read without wait

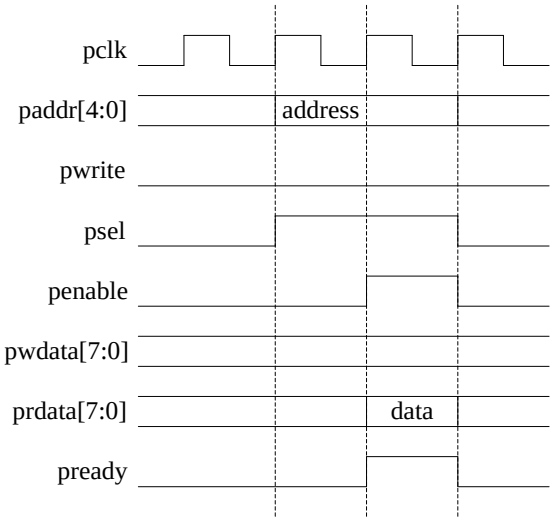

Figure 9-18 Read Without Wait

Read with wait

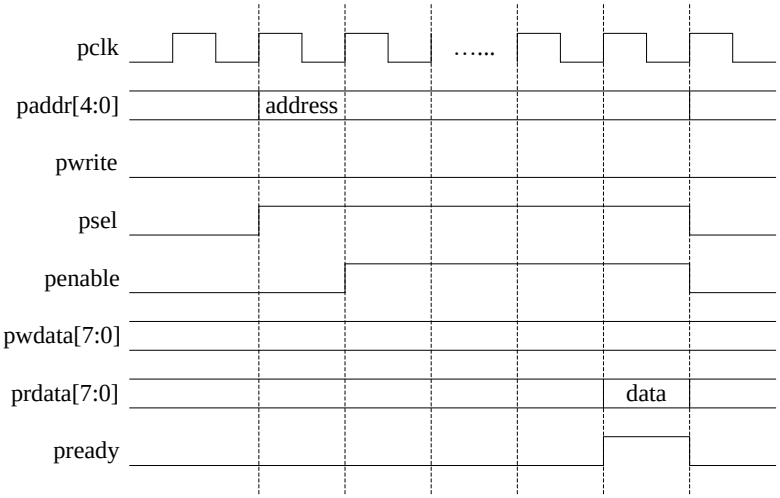

Figure 9-19 Read With Wait

Continuous Read

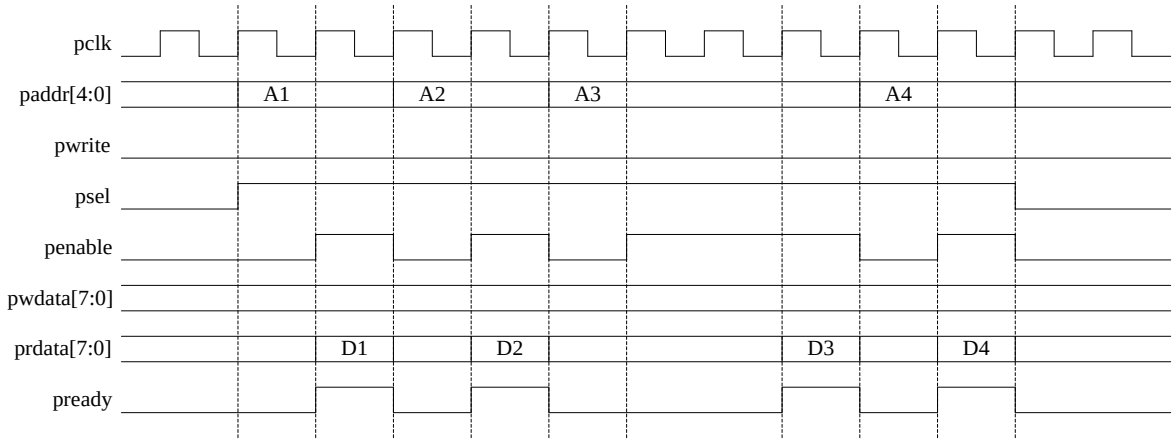

Figure 9-20 Continuous Read

Back-to-Back

Write before read

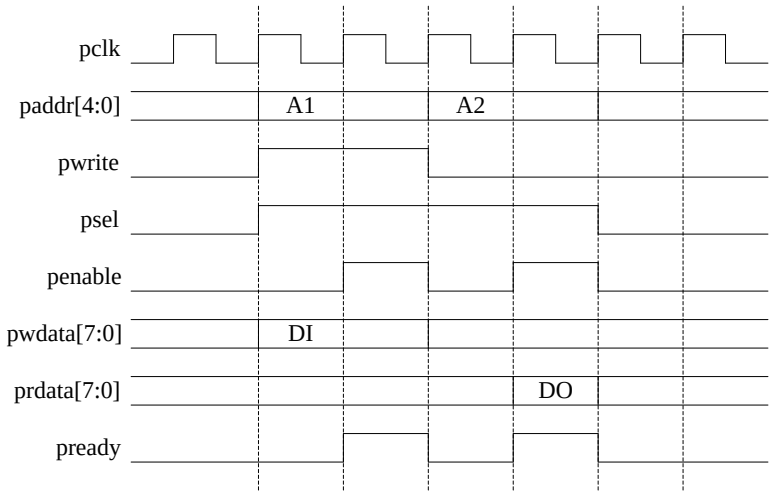

Figure 9-21 Write Before Read

Read before write

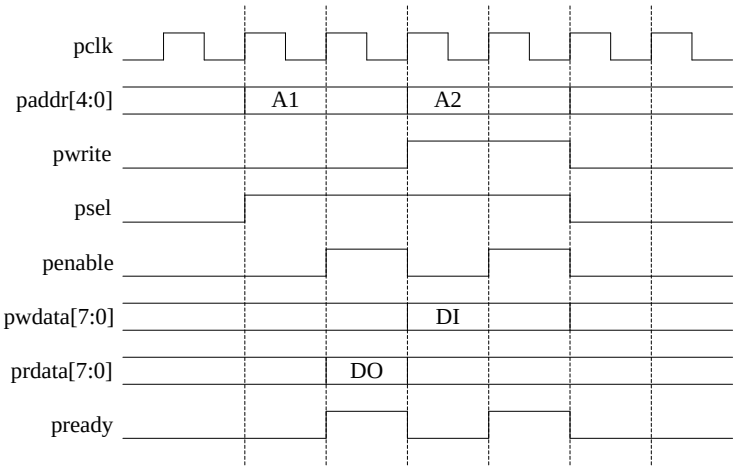

Figure 9-22 Read Before Write

9.5.3 Register

Table 9-8 Internal Slave APB Registers

Item	R/W	Address	Description
IDCODER0	R	00000	IDCODE Register 0
IDCODER1	R	00001	IDCODE Register 1
IDCODER2	R	00010	IDCODE Register 2
IDCODER3	R	00011	IDCODE Register 3
USERCODER0	R	00100	USERCODE Register 0
USERCODER1	R	00101	USERCODE Register 1
USERCODER2	R	00110	USERCODE Register 2
USERCODER3	R	00111	USERCODE Register 3
UIDR0	R	01000	UID Register 0
UIDR1	R	01001	UID Register 1
UIDR2	R	01010	UID Register 2
UIDR3	R	01011	UID Register 3
UIDR4	R	01100	UID Register 4
UIDR5	R	01101	UID Register 5
UIDR6	R	01110	UID Register 6
UIDR7	R	01111	UID Register 7
STATUSR0	R	10000	CCS Status Register 0
STATUSR1	R	10001	CCS Status Register 1
STATUSR2	R	10010	CCS Status Register 2
STATUSR3	R	10011	CCS Status Register 3
IRQCTLR	R/W	10100	Interrupt Control Register
CMDR	R/W	10101	Command register

Item	R/W	Address	Description
ADR0	R/W	10110	Address Register 0
ADR1	R/W	10111	Address Register 1
ADR2	R/W	11000	Address Register 2
DATATR	W	11001	Data Transmit Registers
DATARR	R	11010	Data Receive Registers
STATUSR	R	11011	Status register
IRQSTATUSR	R	11100	Interrupt Status Registers
IRQR	R	11101	Interrupt Registers
LOCKRAR0	R	11110	Lock Page Register 0
LOCKRAR1	R	11111	Lock Page Register 1

Note: For a detailed description of the internal slave APB interface registers, refer to Appendix 6.

9.6 Readback CRC (RBCRC)

Readback CRC uses a 32-bit CRC algorithm, with an initial CRC value of 0. During readback, the 32-bit data undergoes one CRC calculation.

Each readback calculates a CRC value. If the CRC result of the readback calculation is inconsistent with the value in the readback CRC register, the readback CRC error position in the status register is high.

The readback CRC clock is input to both CCS and SRB simultaneously by the OSC.

Figure 9-23 Readback CRC Interface Diagram

9.6.1 Port List

Table 9-9 Readback CRC Port List

Item	I/O	Source/Purpose	Description
RBCRC_CLK	I	OSC	Readback CRC clock, default to 2.08M and with a maximum frequency of 33.3M
RBCRC_RST	I	SRB	Readback CRC Reset Used to reset the readback CRC error flag and clear the readback CRC initial value When readback CRC reset and readback CRC start are both active, readback CRC start cannot be initiated When readback CRC reset is inactive but RBCRC_START

			is active, readback CRC start can be initiated Readback CRC reset can only be performed before the readback CRC start or after the readback CRC ends but cannot be performed during the readback CRC process (from the initiation of the readback CRC start to when the readback CRC information valid flag indicates the readback CRC end)
RBCRC_START	I	SRB	Readback CRC start Pulse-triggered initiation of readback CRC start During the readback CRC process (from the initiation of the readback CRC start to when the readback CRC information valid flag indicates the readback CRC end), the readback CRC start signal is invalid. A new readback CRC initiation can only be started after one readback cycle is completed
RBCRC_ERR	O	SRB	Readback CRC Error Flag
RBCRC_VALID	O	SRB	Readback CRC information valid flag Pulse signal, indicating readback CRC error flag validity and readback completion

9.6.2 Interface timing

9.6.2.1 One-Step Operation

The next readback CRC start follows the previous readback CRC end.

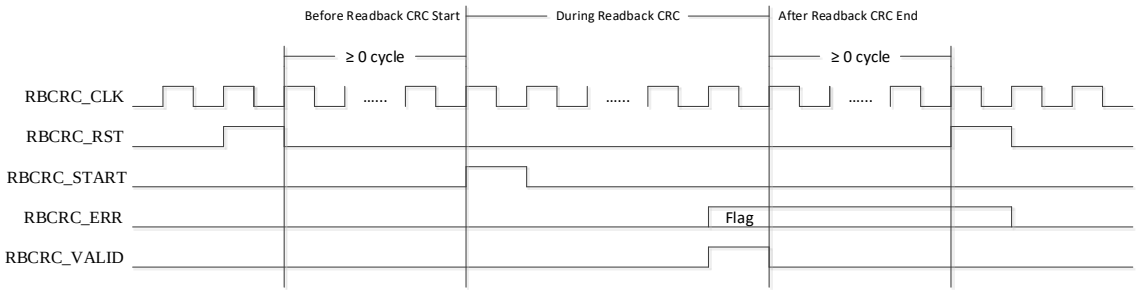

Figure 9-24 One-Step Operation Timing Diagram

9.6.2.2 Continuous Operation

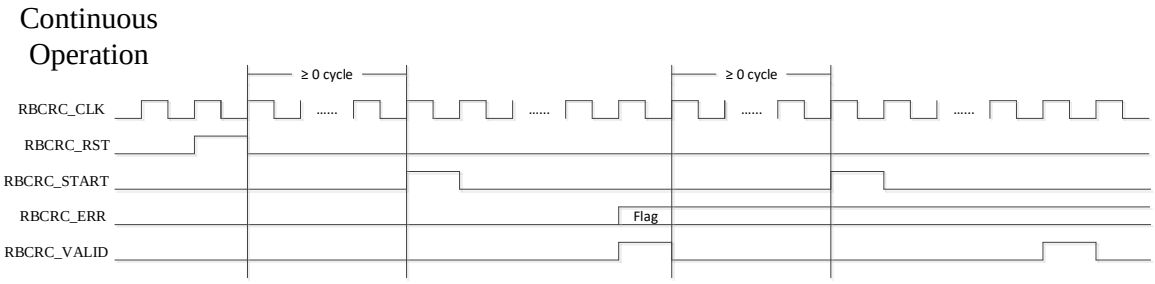

Figure 9-25 Continuous Operation Timing Diagram

9.6.2.3 Burst Operation

During continuous readback CRC, the RBCRC_START signal can be permanently set to 1.

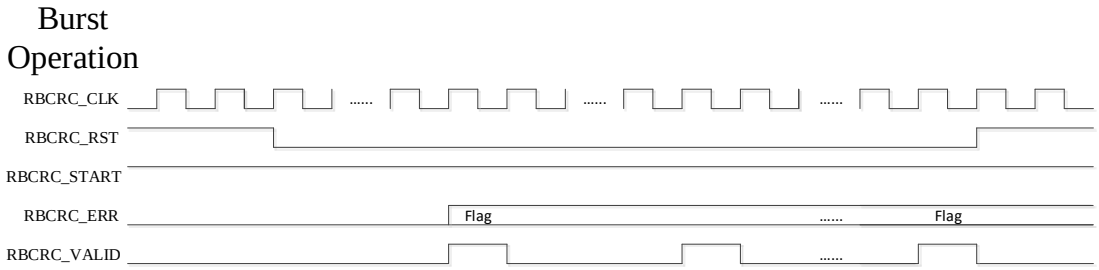

Figure 9-26 One-Step Operation Timing Diagram

Chapter 10 Embedded Hard IP

CPLD devices support SPI, I^2C, and timer features. For more information, please refer to the "***UG030007_Compact Family CPLD Embedded Hard Core User Guide***".

Chapter 11 Appendix 1

JTAG Interface Instruction Description. When writing instructions, start with the least significant bit.

11.1 BYPASS

"Bypass" instruction.

The "Bypass" instruction places the bypass register on the scan chain between TDI and TDO, putting the device in operating mode.

11.2 SAMPLE/PRELOAD

"SAMPLE/PRELOAD" instruction.

The "SAMPLE/PRELOAD" instruction places the boundary scan register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, it captures the data from the device's input pins.

When the TAPC state machine is in the Shift-DR state, it shifts test vectors into the boundary scan register via the scan chain.

When the TAPC state machine is in the Update-DR state, it latches the newly shifted data into the parallel output of the boundary scan cells without updating the device's output pins.

In the "SAMPLE/PRELOAD" instruction mode, the device is in operating mode and can function normally.

11.3 EXTEST

External test instruction.

The "EXTEST" instruction performs interconnect tests between chip pins.

The "EXTEST" instruction places the boundary scan register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, it captures the data from the device's input pins.

When the TAPC state machine is in the Shift-DR state, it shifts the captured data from device input pins out while moving test vectors into the boundary scan register.

When the TAPC state machine is in the Update-DR state, it latches the newly shifted data into the parallel output of the boundary scan cells and updates the device's output pins.

Before the "EXTEST" instruction is executed, a "PRELOAD" instruction operation must be performed to preload test vectors. Once the "EXTEST" instruction takes effect, the preloaded test vectors are driven to the output pins, ensuring controllable output pin states.

In "EXTEST" instruction mode, the device is in test mode and cannot function normally.

11.4 INTEST

Internal test instruction.

The "INTEST" instruction uses the boundary scan register as the device's input and output for conducting static tests of the system logic.

The "INTEST" instruction places the boundary scan register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, it captures data from the device's output pins.

When the TAPC state machine is in the Shift-DR state, it shifts the captured data from the device's output pins out while shifting test vectors into the boundary scan register.

When the TAPC state machine is in the Update-DR state, it latches the newly shifted data into the parallel output of the boundary scan cells and updates the device's input pins.

Before the "INTEST" instruction is executed, a "PRELOAD" instruction operation must be performed to preload test vectors. Once the "INTEST" instruction takes effect, the preloaded test vectors are driven to input pins, ensuring controllable input pin states.

The clock signal is generated in the same manner as non-clock signals, i.e., by shifting in through the boundary scan chain.

11.5 IDCODE

"IDCODE" instruction.

The "IDCODE" instruction reads the device ID code.

The "IDCODE" instruction places the device ID register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the device ID register captures the device ID code on the rising edge of TCK.

When the TAPC state machine is in the Shift-DR state, the device ID register shifts right by one bit on the rising edge of TCK.

11.6 HIGHZ

"HighZ" instruction.

The "HighZ" instruction sets the device's tri-state pin outputs to the High-Z state.

The "HighZ" instruction places the bypass register on the scan chain between TDI and TDO.

11.7 JRST

"Reset" instruction.

The "Reset" instruction resets the systems except the JTAG circuit.

The "USERCODE" instruction places the bypass register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Test/Idle state, the reset operation is executed.

11.8 READ_UID

"Read UID" instruction.

The "Read UID" instruction reads the UID, starting with the least significant bit.

The "Read UID" instruction places the UID register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the UID register captures the UID stored in the eFuse on the rising edge of TCK.

When the TAPC state machine is in the Shift-DR state, on the rising edge of TCK, the UID register shifts right by one bit.

11.9 RDSR

"Read Status Register" instruction.

The "Read Status Register" instruction reads the status register, starting with the least significant bit.

The "Read Status Register" instruction places the status register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the status register captures the value of the CCS status register on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the status register shifts right by one bit on the rising edge of TCK.

11.10 WADR

"Write Address" instruction.

The "Write Address" instruction writes the starting address of the embedded Flash.

The "Write Address" instruction places the address register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the address register captures the starting address of the current embedded Flash on the rising edge of TCK.

When the TAPC state machine is in the data shift state, the address register shifts right by one bit on the rising edge of TCK.

When the TAPC state machine is in the data update state, the data from the shift register path is latched to the parallel output of the address register on the falling edge of TCK.

11.11 ERASE

"Erase" instruction.

The "Erase" instruction erases all normal pages of the embedded Flash and clear the lock flag of the embedded Flash.

The "Erase" instruction places the bypass register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Test/Idle state, the erase operation is executed.

11.12 ERASE_PAGE

"Page Erase" instruction.

The "Page Erase" instruction erases the normal pages of the embedded Flash.

The "Page Erase" instruction places the bypass register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Test/Idle state, the page erase operation is executed.

11.13 ERASE_CTL

"Control Erase" instruction.

The "Control Erase" instruction erases the feature control pages of the embedded Flash.

The "Control Erase" instruction places the bypass register on the scan chain between TDI and TDO.
When the TAPC state machine is in the Test/Idle state, the control erase operation is executed.

11.14 PROGRAM

"Program" instruction.
The "Program" instruction programs the normal pages of the embedded Flash.
The "Program" instruction places the bypass register on the scan chain between TDI and TDO.
When the TAPC state machine is in the Shift-DR state, a page of data is shifted serially into the bypass register on the rising edge of the TCK.

11.15 PROGRAM_CTL

"Control Program" instruction.
The "Control Program" instruction programs the feature control bits of the embedded Flash.
The "Control Program" instruction places the feature control register on the scan chain between TDI and TDO.
When the TAPC state machine is in the Data Capture state, the feature control register captures the feature control bits stored in the embedded Flash on the rising edge of the TCK.
When the TAPC state machine is in the Shift-DR state, the feature control register shifts right by one bit on the rising edge of the TCK.
When the TAPC state machine is in the Update-DR state, data from the shift register path is latched to the parallel output of the feature control register on the falling edge of the TCK.

11.16 READ

"Read" instruction.
The "Read" instruction reads the contents of the embedded Flash, starting with the least significant bit.
The "Read" instruction selects the program register, without placing the program register on the scan chain between TDI and TDO.
When the TAPC state machine is in the Shift-DR state, the ordinary contents of the embedded Flash are shifted out serially through the program register from the TDO on the rising edge of the TCK.

11.17 READ_CTL

"Control Read" instruction.

The "Control Read" instruction reads the feature control bits of the embedded Flash, starting with the least significant bit.

The "Control Program" instruction places the feature control register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the feature control register captures the feature control bits stored in the embedded Flash on the rising edge of the TCK.

When the TAPC state machine is in the Shift-DR state, the feature control register shifts right by one bit on the rising edge of the TCK.

11.18 PROGRAM_LOCK

"Lock" instruction.

The "Lock" instruction programs the lock flag and the bitstream tail page address into the embedded Flash, preventing program and read operations on the bitstream.

The "Lock" instruction places the lock register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the lock register captures the lock flag and bitstream tail page address on the rising edge of the TCK.

When the TAPC state machine is in the data shift state, the LOCK register shifts right by one bit on the rising edge of TCK.

When the TAPC state machine is in the Test/Idle state, the program lock flag operation is executed, programming the lock flag and bitstream tail page address into the embedded Flash.

11.19 READ_LOCK

"Read Lock Flag" instruction.

The "Read Lock Flag" instruction reads the lock flag and the bitstream tail page address.

The "Read Lock Flag" instruction places the lock register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Data Capture state, the lock register captures the lock flag and bitstream tail page address on the rising edge of the TCK.

When the TAPC state machine is in the data shift state, the LOCK register shifts right by one bit on the rising edge of TCK.

11.20 EFlash_SLEEP

"Embedded Flash Sleep" instruction.

The "Embedded Flash Sleep" instruction puts the embedded Flash into sleep mode.

The "Embedded Flash Sleep" instruction places the bypass register on the scan chain between TDI and TDO.

The embedded Flash sleep operation is executed when the TAPC state machine is in the Test/Idle state.

11.21 EFlash_WAKEUP

"Embedded Flash Wake-up" instruction.

The "Embedded Flash Wake-up" instruction wakes up the embedded Flash.

The "Embedded Flash Wake-up" instruction places the bypass register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Test/Idle state, the wake-up operation of the embedded Flash is executed.

11.22 CFGI

"Configuration" instruction.

The "Configuration" instruction configures the contents of the configuration memory and configuration register.

The "Configuration" instruction places the bypass register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Shift-DR state, the bit stream is serially shifted into the bypass register on the rising edge of TCK.

11.23 CFGO

"Readback" instruction.

The "Readback" instruction reads back the contents of the configuration memory and configuration register, starting with the least significant bit.

The "Readback" instruction selects the configuration memory but does not place it on the scan chain between TDI and TDO.

When the TAPC state machine is in the Shift-DR state, on the rising edge of TCK, the contents of the configuration memory and configuration register are serially shifted out through the configuration memory from TDO.

11.24 JWAKEUP

"Wake-up" instruction.

The "Wake-up" instruction wakes up devices.

The "Wake-up" instruction places the bypass register on the scan chain between TDI and TDO.

When the TAPC state machine is in the Test/Idle state, the wake-up operation is executed.

Chapter 12 Appendix 2

JTAG Interface Operation Process Description. For the specific implementation of the C code in the following process, please refer to the application guide "AN03008 Compact Family CPLD Device JTAG Interface Configuration and Programming Guide", as well as the corresponding example project.

In the following process, the number of continuous clock cycles in the Run Test Idle state is determined based on a fixed delay time with TCK at 50MHz; users can determine the number of clocks based on the actual TCK frequency. Users can determine the number of clocks to wait based on the actual TCK frequency. If TCK is at 50MHz, 100 cycles are needed; if at 5MHz, 10 clock cycles are required.

12.1 Scan Chain

Table 12-1 Scan Chain

Steps	Operation Description
1	Load "IDCODE" instruction into IR
2	Shift out IDCODE in the Shift-DR state, starting with the least significant bit
3	Enter the Test Logic Reset state

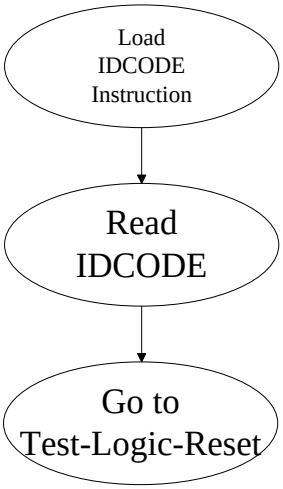

Figure 12-1 Scan Chain

12.2 Read the status register

Table 12-2 Read Status Register

Steps	Operation Description

Steps	Operation Description
1	Load RDSR instruction into IR
2	Shift out the value of the status register in the Shift-DR state, starting with the least significant bit
3	Enter the Test Logic Reset state

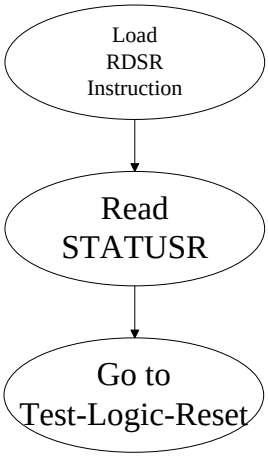

Figure 12-2 Read Status Register

12.3 Put embedded Flash into Sleep Mode

This operation can only be initiated when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-3 Put embedded Flash into Sleep Mode

Steps	Operation Description
1	Load the "EFlash_SLEEP" instruction into IR
2	Enter the Run Test Idle state and continue for at least 12,000 clock cycles
3	Load RDSR instruction into IR
4	Shift out the contents of the status register in the Shift-DR state, starting with the least significant bit
5	Enter the Test Logic Reset state

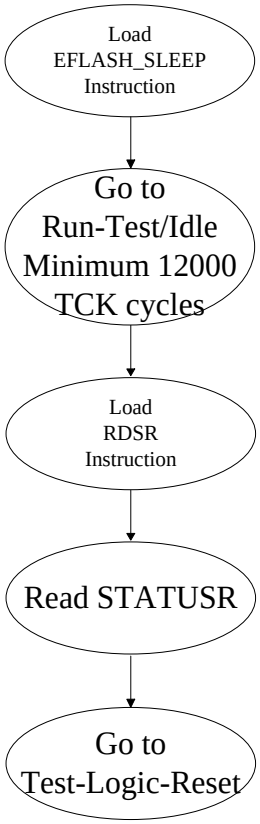

Figure 12-3 Put embedded Flash into Sleep Mode

12.4 Wake Up embedded Flash

This operation can only be initiated when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-4 Wake Up embedded Flash

Steps	Operation Description
1	Load the "EFlash_WAKEUP" instruction into IR
2	Enter the Run Test Idle state and continue for at least 20,000 clock cycles
3	Load RDSR instruction into IR
4	Shift out the contents of the status register in the Shift-DR state, starting with the least significant bit
5	Enter the Test Logic Reset state

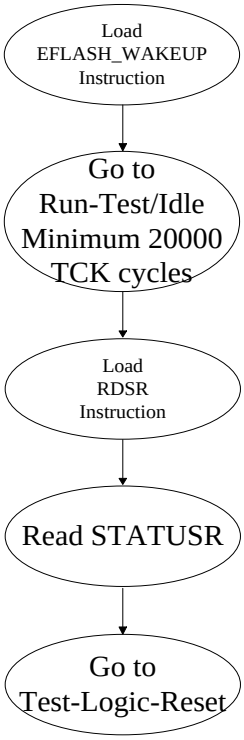

Figure 12-4 Wake Up embedded Flash

12.5 Read UID

Table 12-5 Read UID

Steps	Operation Description
1	Load the "READ_UID" instruction into IR
2	Shift out UID in the Shift-DR state, starting with the least significant bit
3	Enter the Test Logic Reset state

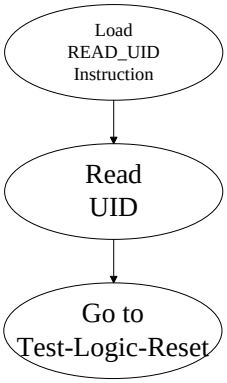

Figure 12-5 Read UID

12.6 Program Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-6 Program Feature Control Bits

Steps	Operation Description
1	Load the "PROGRAM_CTL" instruction into IR
2	Shift in the feature control bits in the Shift-DR state, starting with the least significant bit
3	Enter the Run Test Idle state and continue for at least 450,000 clock cycles (600,000 clock cycles are required for the 10KD model)
4	Enter the Test Logic Reset state
5	Load the "READ_CTL" instruction into IR
6	Enter the Run Test Idle state and continue for at least 50 clock cycles
7	Shift out the feature control bits in the Shift-DR state, starting with the least significant bit
8	Enter the Test Logic Reset state

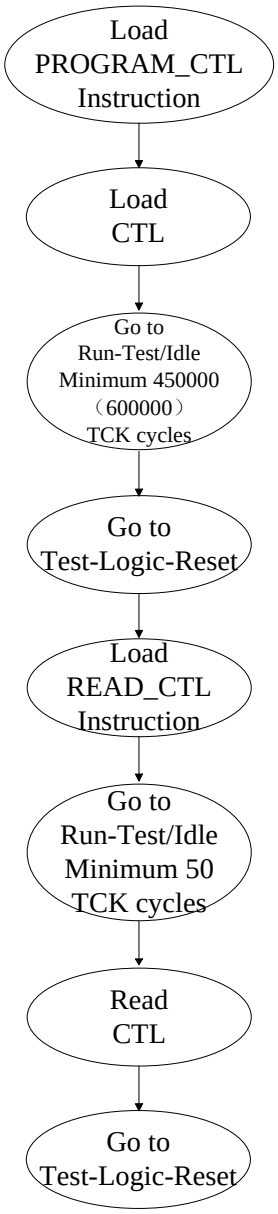

Figure 12-6 Program Feature Control Bits

12.7 Reset Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-7 Reset Feature Control Bits

Steps	**Operation Description**
1	Load the "ERASE_CTL" instruction into IR
2	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
3	Enter the Test Logic Reset state
4	Load the "READ_CTL" instruction into IR
5	Enter the Run Test Idle state and continue for at least 50 clock cycles

Steps	Operation Description
6	Shift out the feature control bits in the Shift-DR state, starting with the least significant bit
7	Enter the Test Logic Reset state

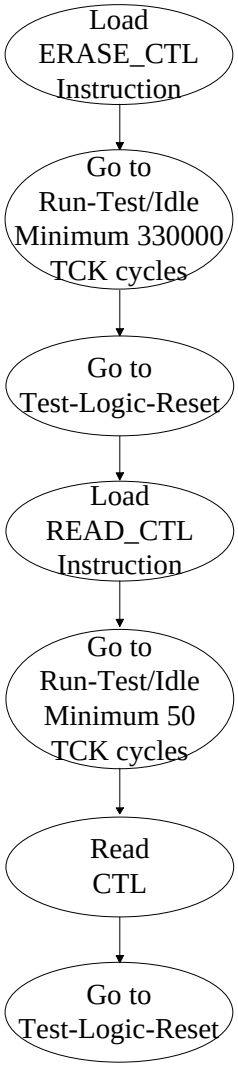

Figure 12-7 Reset Feature Control Bits

12.8 Read Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-8 Read Feature Control Bits

Steps	Operation Description
1	Load the "READ_CTL" instruction into IR
2	Enter the Run Test Idle state and continue for at least 50 clock cycles
3	Shift out the feature control bits in the Shift-DR state, starting with the least significant bit
4	Enter the Test Logic Reset state

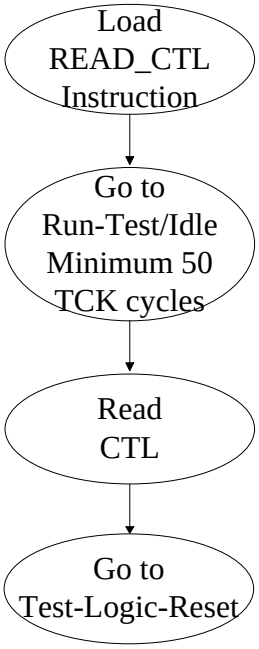

Figure 12-8 Read Feature Control Bits

12.9 Program Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

12.9.1 1K/2K/4K/7K

Table 12-9 Program Bitstream

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the address of the first page of the data stream from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "PROGRAM" instruction into IR
4	Shift in the first page of the data stream in the Shift-DR state by shifting in 32 bits each time, starting with the most significant bit
5	Enter the Run Test Idle state and continue for at least 470,000 clock cycles
6	Load the "READ" instruction into IR
7	Enter the Run Test Idle state and continue for at least 50 clock cycles
8	Shift out the first page of the data stream in the Shift-DR state, starting with the most significant bit
9	Load the "WADR" instruction into IR
10	Shift in the second page of the data stream from the embedded Flash in the Shift-DR state, starting with the least significant bit
11	Load the "PROGRAM" instruction into IR
12	Shift in the second page of the data stream in the Shift-DR state by shifting in 32 bits each time, starting with the most significant bit
13	Enter the Run Test Idle state and continue for at least 470,000 clock cycles
14	Load the "READ" instruction into IR

Steps	Operation Description
15	Enter the Run Test Idle state and continue for at least 50 clock cycles
16	Shift out the second page of the data stream in the Shift-DR state, starting with the most significant bit
17	……
18	Load the "WADR" instruction into IR
19	Shift in the nth page of the data stream from the embedded Flash in the Shift-DR state, starting with the least significant bit
20	Load the "PROGRAM" instruction into IR
21	Shift in the nth page of the data stream in the Shift-DR state by shifting in 32 bits each time, starting with the most significant bit
22	Enter the Run Test Idle state and continue for at least 470,000 clock cycles
23	Load the "READ" instruction into IR
24	Enter the Run Test Idle state and continue for at least 50 clock cycles
25	Shift out the nth page of the data stream in the Shift-DR state, starting with the most significant bit
26	Enter the Test Logic Reset state

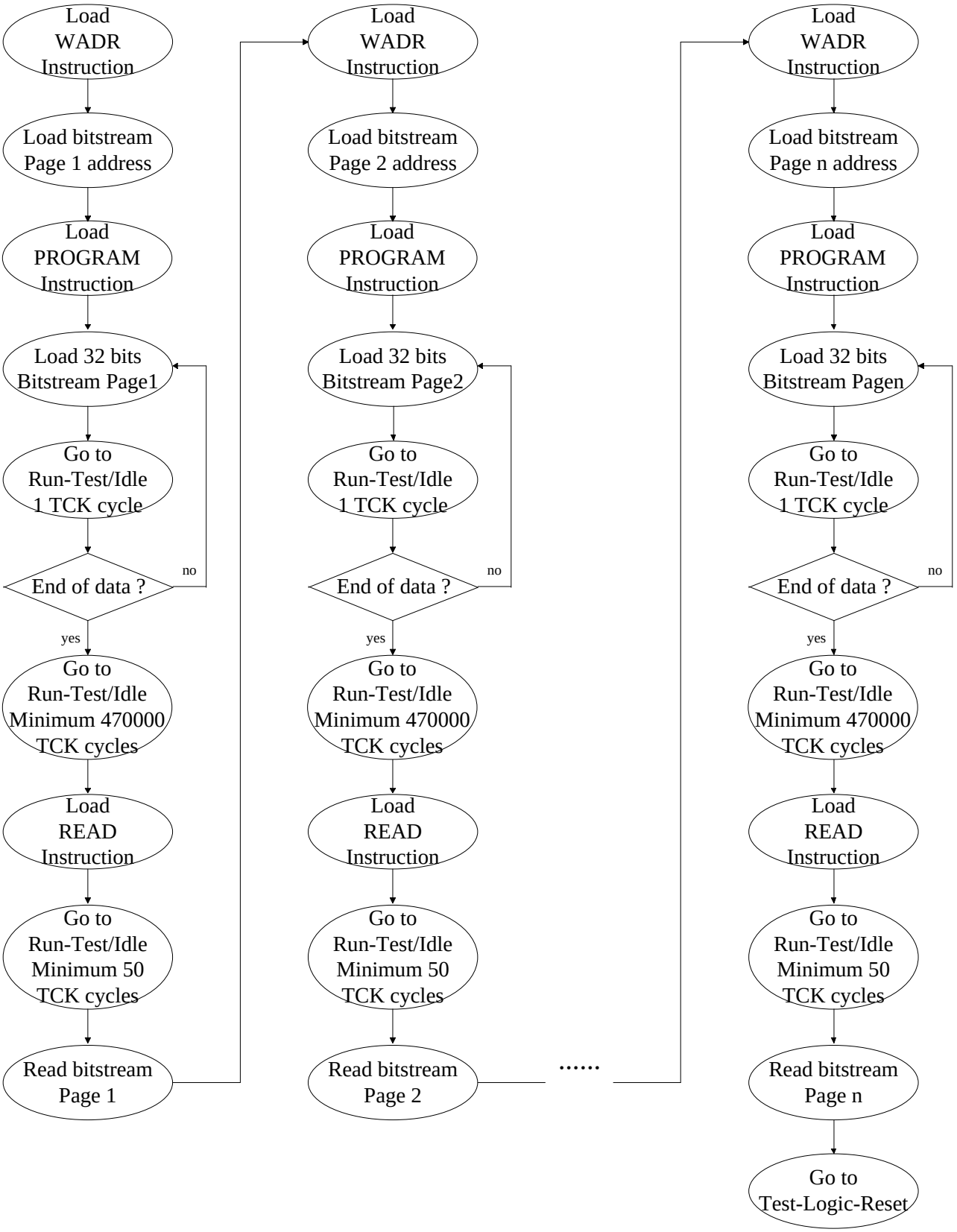

Figure 12-9 Program Bitstream

12.9.2 10K

Table 12-10 Program Bitstream

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the address of the first page of the data stream from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "PROGRAM" instruction into IR
4	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
5	Shift in the first page of the data stream in the Shift-DR state by shifting in 32 bits each time, starting with the most significant bit
6	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
7	Load the "READ" instruction into IR
8	Enter the Run Test Idle state and continue for at least 50 clock cycles
9	Shift out the first page of the data stream in the Shift-DR state, starting with the most significant bit
10	Load the "WADR" instruction into IR
11	Shift in the second page of the data stream from the embedded Flash in the Shift-DR state, starting with the least significant bit
12	Load the "PROGRAM" instruction into IR
13	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
14	Shift in the second page of the data stream in the Shift-DR state by shifting in 32 bits each time, starting with the most significant bit
15	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
16	Load the "READ" instruction into IR
17	Enter the Run Test Idle state and continue for at least 50 clock cycles
18	Shift out the second page of the data stream in the Shift-DR state, starting with the most significant bit
19	……
20	Load the "WADR" instruction into IR
21	Shift in the nth page of the data stream from the embedded Flash in the Shift-DR state, starting with the least significant bit
22	Load the "PROGRAM" instruction into IR
23	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
24	Shift in the nth page of the data stream in the Shift-DR state by shifting in 32 bits each time, starting with the most significant bit
25	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
26	Load the "READ" instruction into IR
27	Enter the Run Test Idle state and continue for at least 50 clock cycles
28	Shift out the nth page of the data stream in the Shift-DR state, starting with the most significant bit
29	Enter the Test Logic Reset state

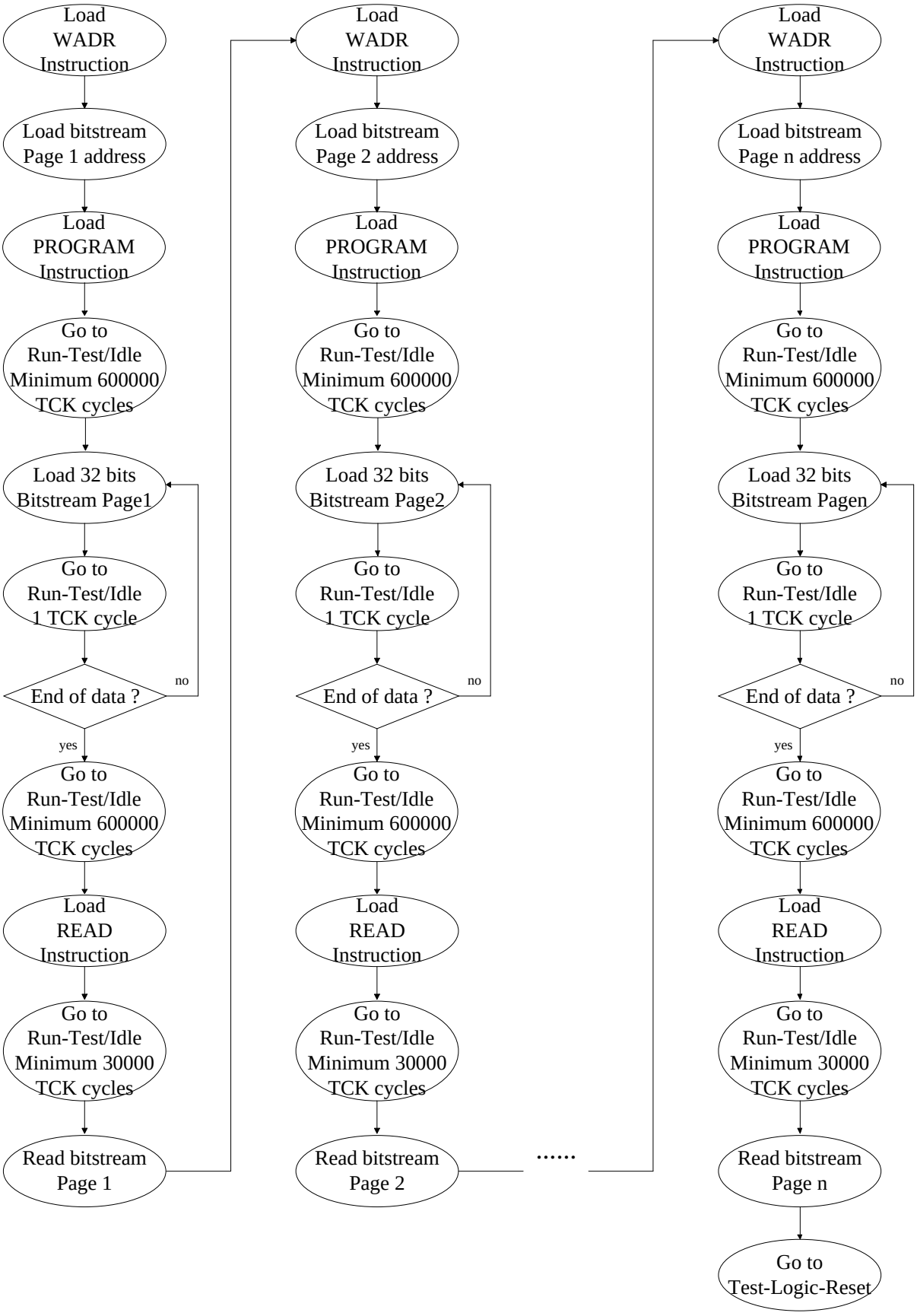

Figure 12-10 Program Bitstream

12.10 Erase Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-11 Erase Bitstream

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the address of the first page of the bitstream data from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "ERASE_PAGE" instruction into IR
4	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
5	Enter the Test Logic Reset state
6	Load the "READ" instruction into IR
7	Enter the Run Test Idle state and continue for at least 50 clock cycles
8	Shift out the first page of the bitstream data in the Shift-DR state, starting with the most significant bit
9	Load the "WADR" instruction into IR
10	Shift in the address of the second page of the bitstream data from the embedded Flash in the Shift-DR state, starting with the least significant bit
11	Load the "ERASE_PAGE" instruction into IR
12	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
13	Enter the Test Logic Reset state
14	Load the "READ" instruction into IR
15	Enter the Run Test Idle state and continue for at least 50 clock cycles
16	Shift out the second page of the bitstream data in the Shift-DR state, starting with the most significant bit
17	……
18	Load the "WADR" instruction into IR
19	Shift in the address of the nth page of the bitstream data from the embedded Flash in the Shift-DR state, starting with the least significant bit
20	Load the "ERASE_PAGE" instruction into IR
21	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
22	Enter the Test Logic Reset state
23	Load the "READ" instruction into IR
24	Enter the Run Test Idle state and continue for at least 50 clock cycles
25	Shift out the nth page of the bitstream data in the Shift-DR state, starting with the most significant bit
26	Enter the Test Logic Reset state

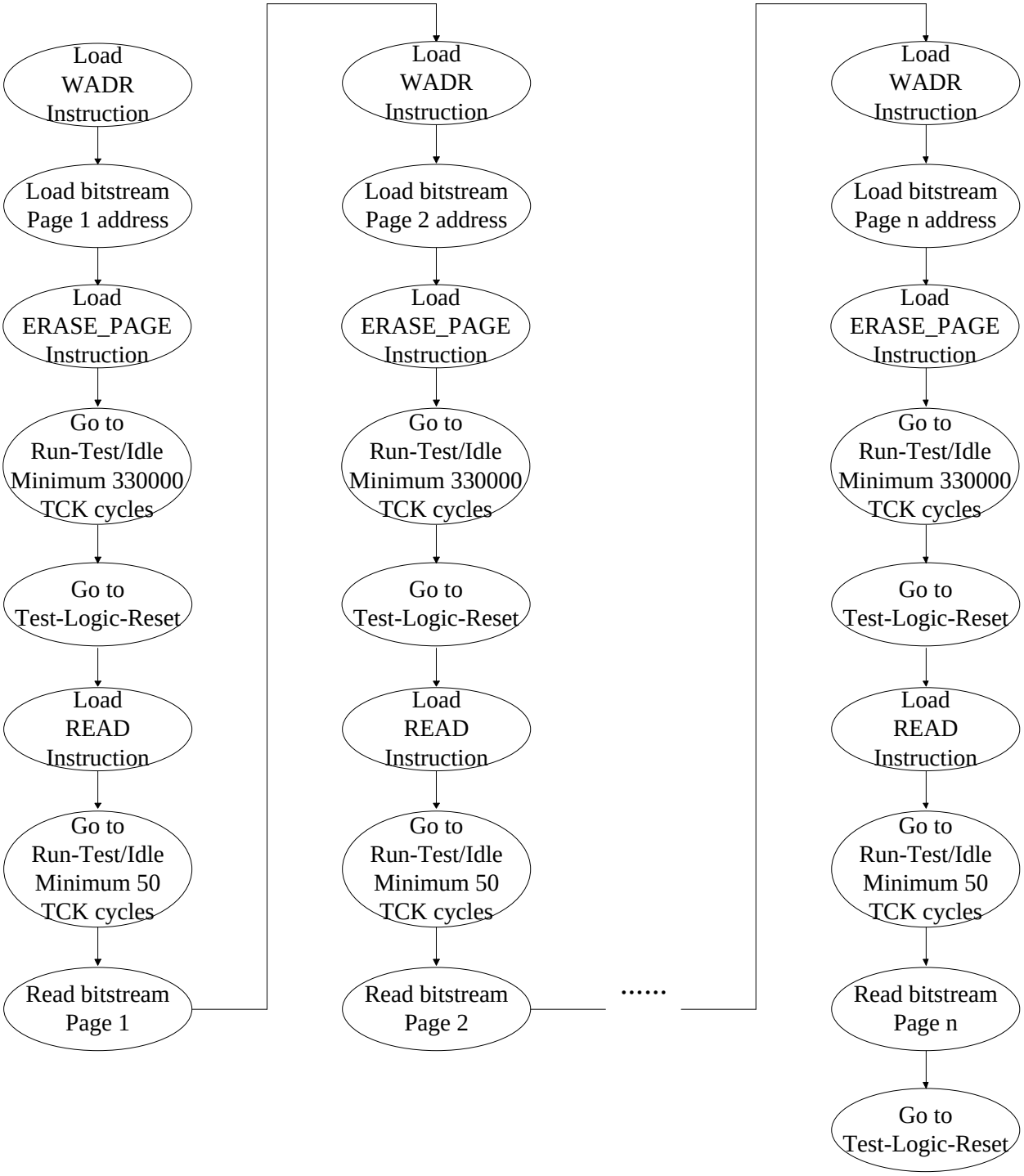

Figure 12-11 Erase Bitstream

12.11 Read Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-12 Read Bitstream

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift the starting address of the bitstream from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "READ" instruction into IR
4	Enter the Run Test Idle state and continue for at least 50 clock cycles
5	Shift out the bitstream in the Shift-DR state, starting with the most significant bit
6	Enter the Test Logic Reset state

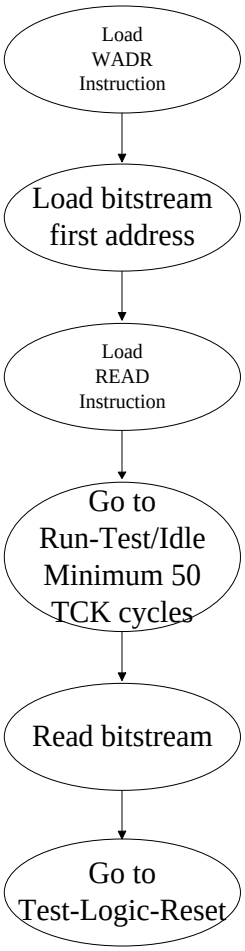

Figure 12-12 Read Bitstream

12.12 Program User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

12.12.1 1K/2K/4K/7K

Table 12-13 Program User Flash

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the address of the first page of the user Flash data from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "PROGRAM" instruction into IR
4	Shift in the first page of the user Flash data in the Shift-DR state by shifting 32 bits each time, starting with the most significant bit
5	Enter the Run Test Idle state and continue for at least 470,000 clock cycles
6	Load the "READ" instruction into IR
7	Enter the Run Test Idle state and continue for at least 50 clock cycles
8	Shift out the first page of the user Flash data in the Shift-DR state, starting with the most significant bit
9	Load the "WADR" instruction into IR
10	Shift in the address of the second page of the user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
11	Load the "PROGRAM" instruction into IR
12	Shift in the second page of the user Flash data in the Shift-DR state by shifting 32 bits at a time, starting with the most significant bit
13	Enter the Run Test Idle state and continue for at least 470,000 clock cycles
14	Load the "READ" instruction into IR
15	Enter the Run Test Idle state and continue for at least 50 clock cycles
16	Shift out the second page of the user Flash data in the Shift-DR state, starting with the most significant bit
17	……
18	Load the "WADR" instruction into IR
19	Shift in the address of the nth page of the user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
20	Load the "PROGRAM" instruction into IR
21	Shift in the nth page of the user Flash data in the Shift-DR state by shifting 32 bits at a time, starting with the most significant bit
22	Enter the Run Test Idle state and continue for at least 470,000 clock cycles
23	Load the "READ" instruction into IR
24	Enter the Run Test Idle state and continue for at least 50 clock cycles
25	Shift out the nth page of the user Flash data in the Shift-DR state, starting with the most significant bit
26	Enter the Test Logic Reset state

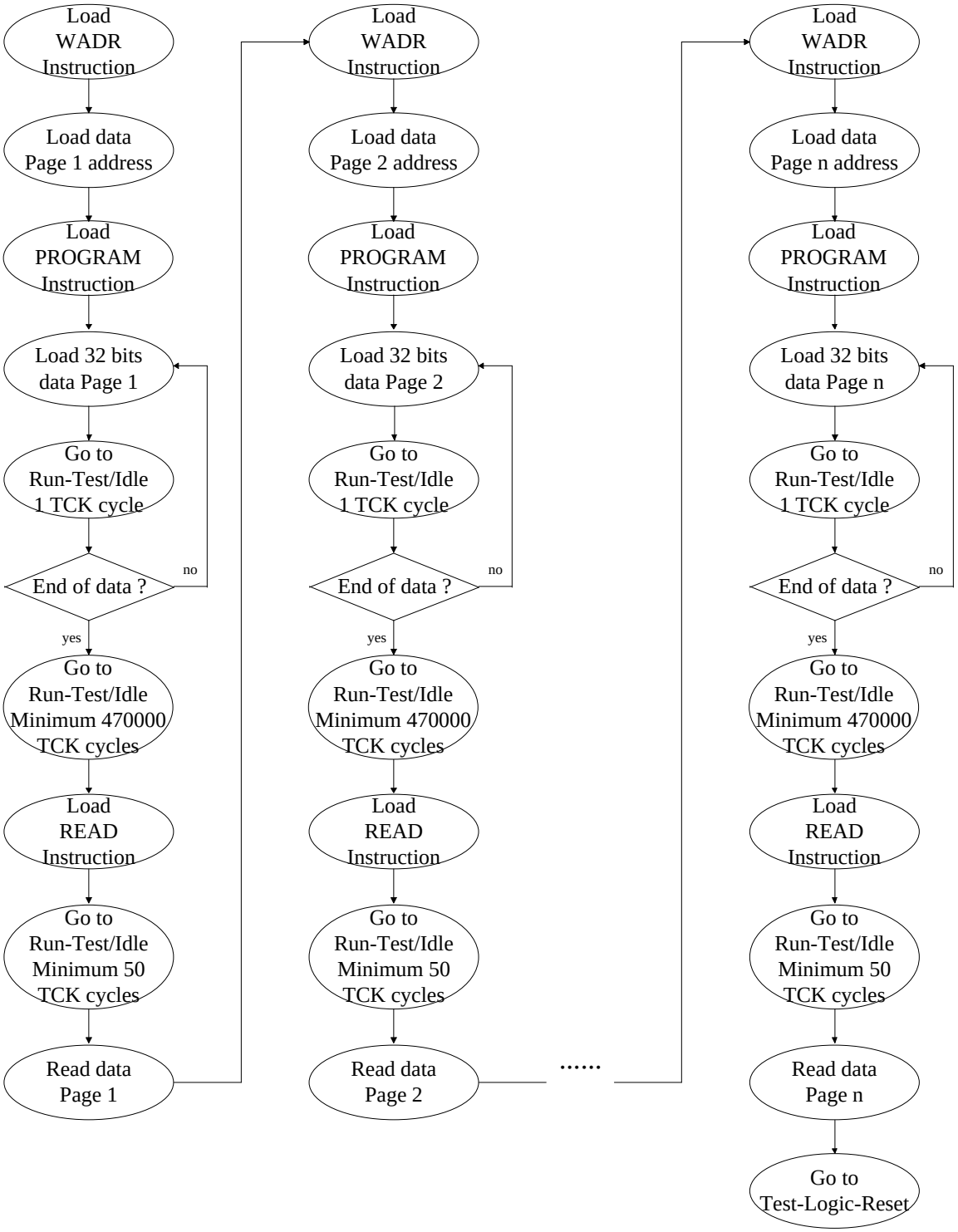

Figure 12-13 Program User Flash

12.12.2 10K

Table 12-14 Program User Flash

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the address of the first page of the user Flash data from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "PROGRAM" instruction into IR
4	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
5	Shift in the first page of the user Flash data in the Shift-DR state by shifting 32 bits each time, starting with the most significant bit
6	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
7	Load the "READ" instruction into IR
8	Enter the Run Test Idle state and continue for at least 50 clock cycles
9	Shift out the first page of the user Flash data in the Shift-DR state, starting with the most significant bit
10	Load the "WADR" instruction into IR
11	Shift in the address of the second page of the user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
12	Load the "PROGRAM" instruction into IR
13	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
14	Shift in the second page of the user Flash data in the Shift-DR state by shifting 32 bits at a time, starting with the most significant bit
15	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
16	Load the "READ" instruction into IR
17	Enter the Run Test Idle state and continue for at least 50 clock cycles
18	Shift out the second page of the user Flash data in the Shift-DR state, starting with the most significant bit
19	……
20	Load the "WADR" instruction into IR
21	Shift in the address of the nth page of the user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
22	Load the "PROGRAM" instruction into IR
23	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
24	Shift in the nth page of the user Flash data in the Shift-DR state by shifting 32 bits at a time, starting with the most significant bit
25	Enter the Run Test Idle state and continue for at least 600,000 clock cycles
26	Load the "READ" instruction into IR
27	Enter the Run Test Idle state and continue for at least 50 clock cycles
28	Shift out the nth page of the user Flash data in the Shift-DR state, starting with the most significant bit
29	Enter the Test Logic Reset state

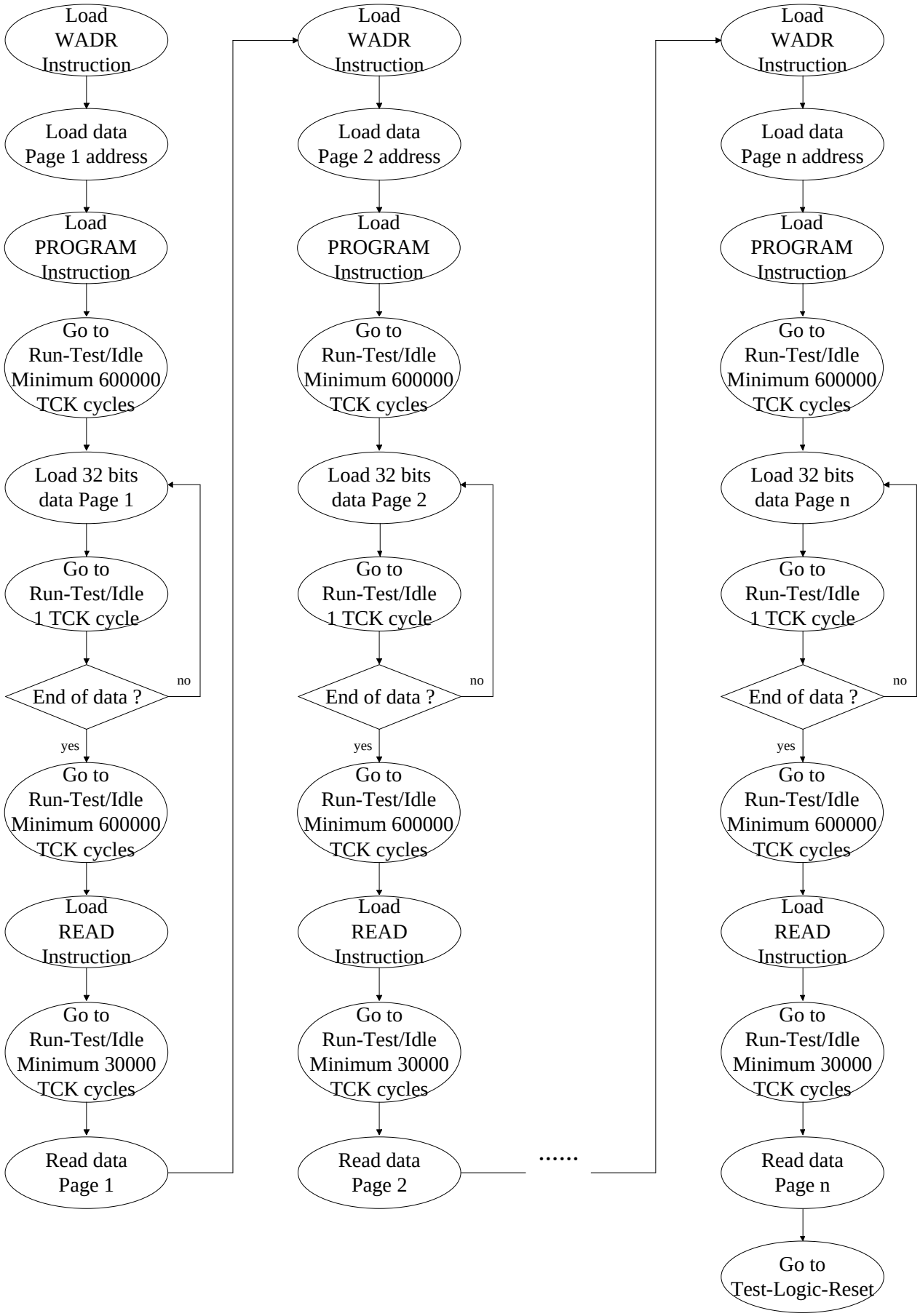

Figure 12-14 Program User Flash

12.13 Erase User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-15 Erase User Flash

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the address of the first page of the user Flash data from the embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "ERASE_PAGE" instruction into IR
4	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
5	Enter the Test Logic Reset state
6	Load the "READ" instruction into IR
7	Enter the Run Test Idle state and continue for at least 50 clock cycles
8	Shift out the first page of the user Flash data in the Shift-DR state, starting with the most significant bit
9	Load the "WADR" instruction into IR
10	Shift in the address of the second page of the user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
11	Load the "ERASE_PAGE" instruction into IR
12	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
13	Enter the Test Logic Reset state
14	Load the "READ" instruction into IR
15	Enter the Run Test Idle state and continue for at least 50 clock cycles
16	Shift out the second page of the user Flash data in the Shift-DR state, starting with the most significant bit
17	……
18	Load the "WADR" instruction into IR
19	Shift in the address of the nth page of the user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
20	Load the "ERASE_PAGE" instruction into IR
21	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
22	Enter the Test Logic Reset state
23	Load the "READ" instruction into IR
24	Enter the Run Test Idle state and continue for at least 50 clock cycles
25	Shift out the nth page of the user Flash data in the Shift-DR state, starting with the most significant bit
26	Enter the Test Logic Reset state

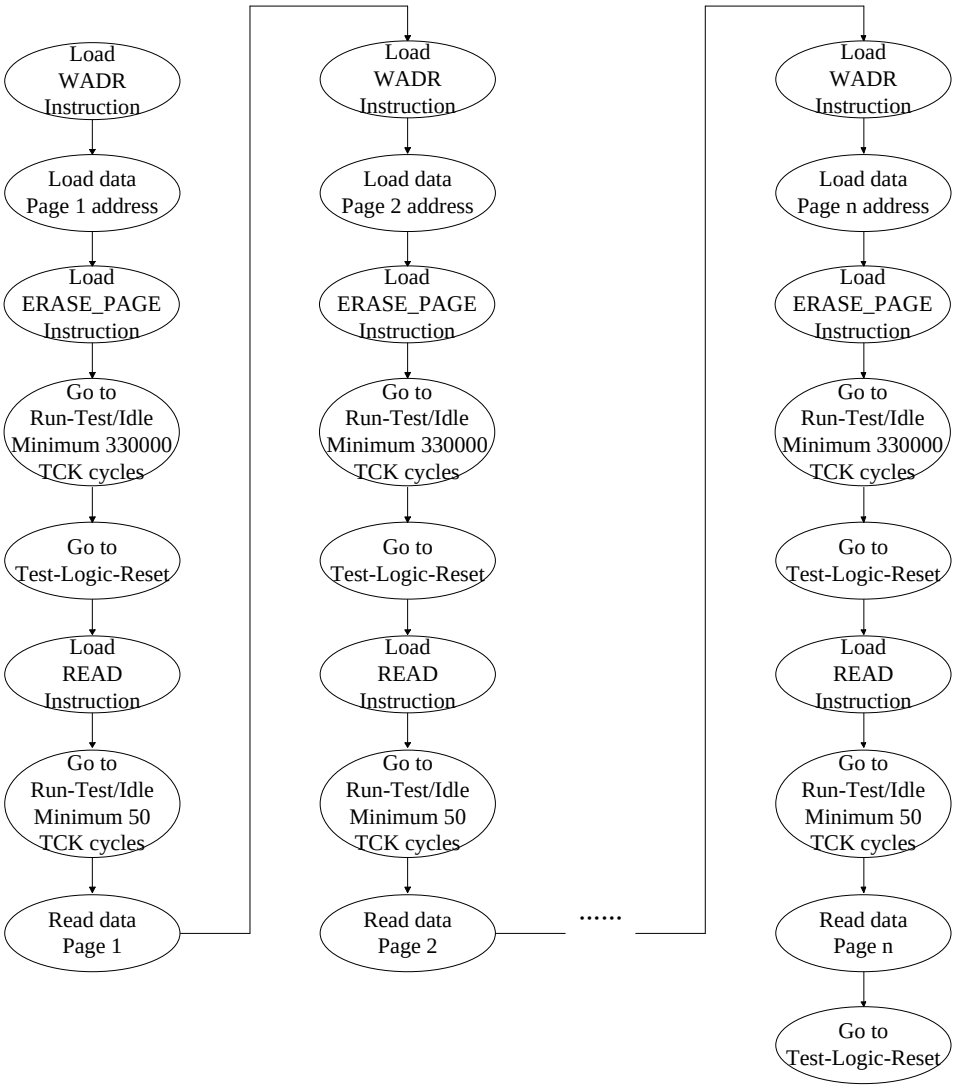

Figure 12-15 Erase User Flash

12.14 Erase Bitstream and User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-16 Erase Bitstream and User Flash

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the starting address of bitstream and user Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "ERASE" instruction into IR
4	Enter the Run Test Idle state and continue for at least 330,000 clock cycles
5	Enter the Test Logic Reset state
6	Load the "READ" instruction into IR
7	Enter the Run Test Idle state and continue for at least 50 clock cycles
8	Shift out the bitstream and user Flash data in the Shift-DR state, starting with the most significant bit

Steps	Operation Description
9	Enter the Test Logic Reset state

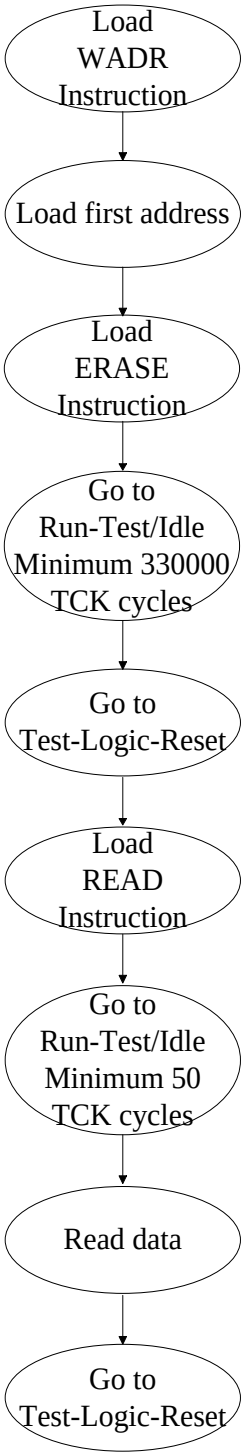

Figure 12-16 Erase Bitstream and User Flash

12.15 Read User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-17 Read User Flash

Steps	Operation Description
1	Load the "WADR" instruction into IR
2	Shift in the starting address of the User Flash data from embedded Flash in the Shift-DR state, starting with the least significant bit
3	Load the "READ" instruction into IR
4	Enter the Run Test Idle state and continue for at least 50 clock cycles
5	Shift out the user Flash data in the Shift-DR state, starting with the most significant bit
6	Enter the Test Logic Reset state

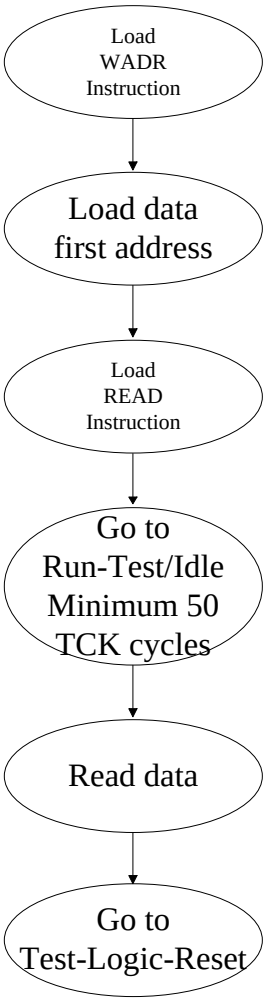

Figure 12-17 Read User Flash

12.16 Lock embedded Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 12-18 Lock embedded Flash

Steps	Operation Description
1	Load the "PROGRAM_LOCK" instruction into IR
2	Shift in the bitstream tail page address and the embedded Flash lock flag bit in the Shift-DR state, starting with the least significant bit
3	Enter the Run Test Idle state and continue for at least 450,000 clock cycles (600,000 clock cycles are required for the 10KD model)
4	Enter the Test Logic Reset state
5	Load the "READ_LOCK" instruction into IR
6	Shift out the bitstream tail page address and the embedded Flash lock flag bit in the Shift-DR state, starting with the least significant bit
7	Enter the Test Logic Reset state

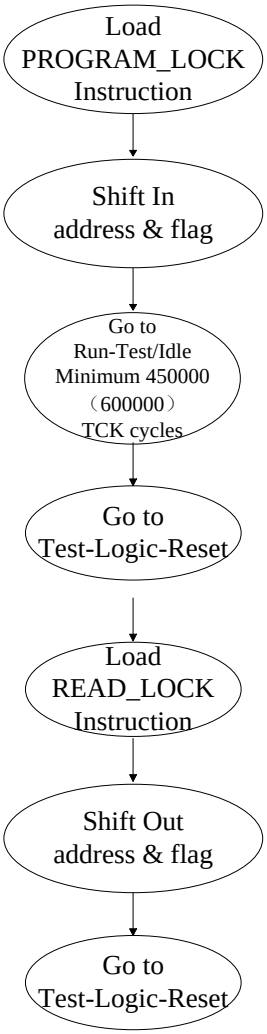

Figure 12-18 Lock embedded Flash

Chapter 13 Appendix 3

Description of Slave SPI/ Slave I^2C interface operation process. For the specific implementation of the C code in the following process, please refer to the "AN03009 Compact Family CPLD Device SPI Interface Configuration and Programming Guide" and "AN03010 Compact Family CPLD Device I^2C Interface Configuration and Programming Guide", along with the corresponding sample projects.

13.1 Read UID

Table 13-1 Read UID

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "RDUID" instruction
3	Write the "WRDIS" instruction

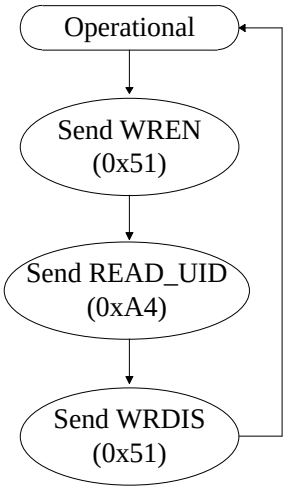

Figure 13-1 Read UID

13.2 Put embedded Flash into Sleep Mode

This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-2 Put embedded Flash into Sleep Mode

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "EFlash_SLEEP" instruction
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "RDSR" instruction
5	Write the "WRDIS" instruction

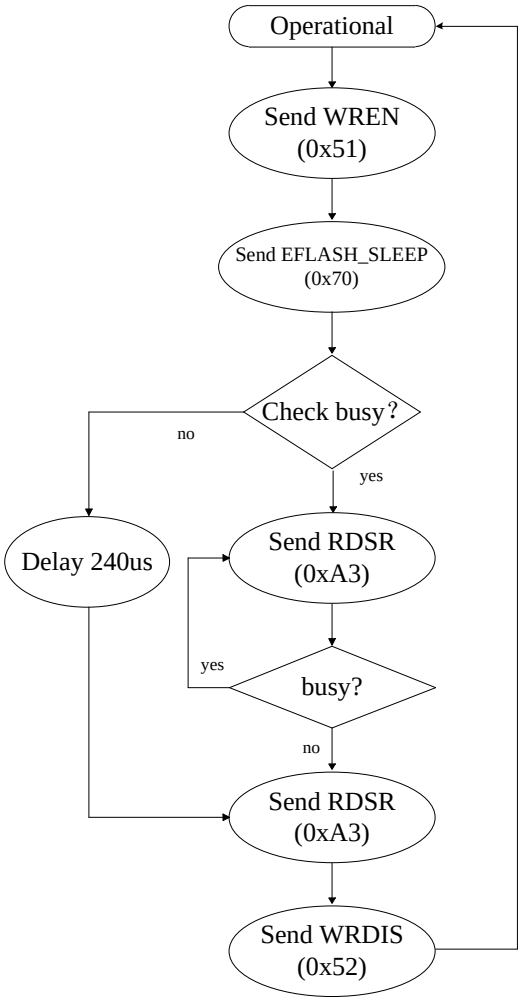

Figure 13-2 Put embedded Flash into Sleep Mode

13.3 Wake Up embedded Flash

This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-3 Wake Up embedded Flash

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "EFlash_WAKEUP" instruction
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "RDSR" instruction
5	Write the "WRDIS" instruction

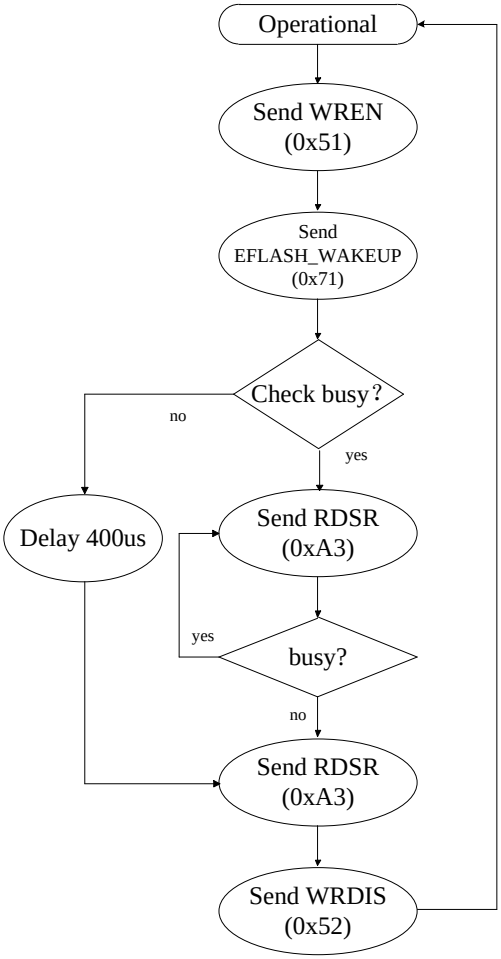

Figure 13-3 Wake Up embedded Flash

13.4 Program Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-4 Program Feature Control Bits

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "PROGRAM_CTL" instruction
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ_CTL" instruction
5	Write the "WRDIS" instruction

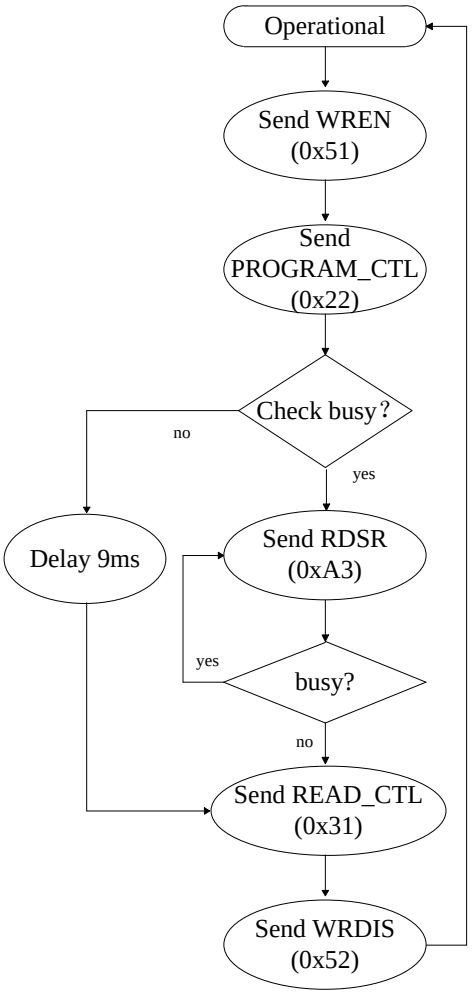

Figure 13-4 Program Feature Control Bits

13.5 Reset Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-5 Reset Feature Control Bits

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "ERASE_CTL" instruction
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ_CTL" instruction
5	Write the "WRDIS" instruction

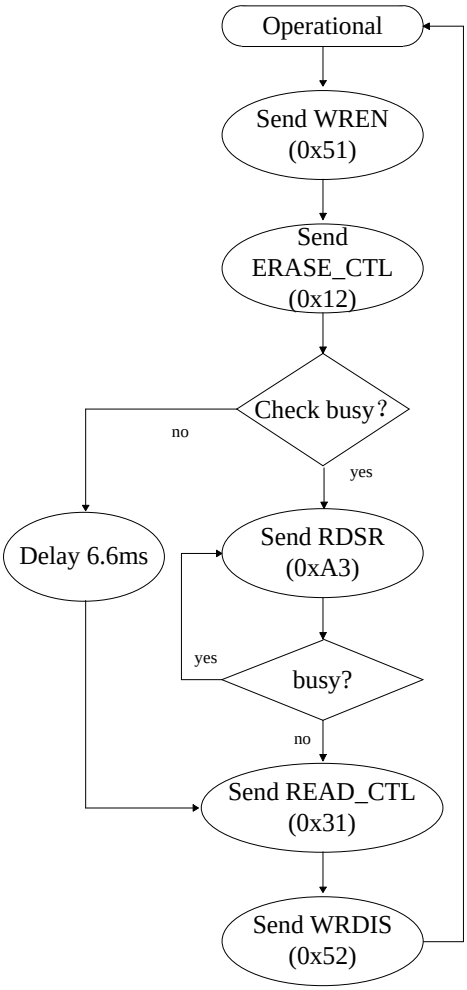

Figure 13-5 Reset Feature Control Bits

13.6 Read Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-6 Read Feature Control Bits

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "READ_CTL" instruction
3	Write the "WRDIS" instruction

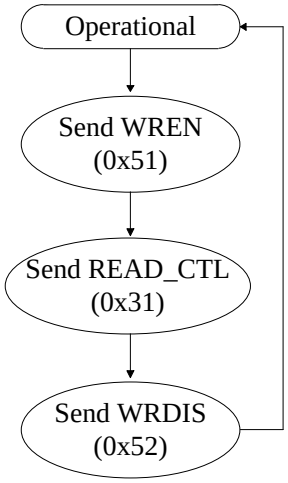

Figure 13-6 Read Feature Control Bits

13.7 Program Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-7 Program Bitstream

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "PROGRAM" instruction for programming the first page of the bitstream data
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ" instruction for reading the first page of the bitstream data
5	Write the "PROGRAM" instruction for programming the second page of the bitstream data
6	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
7	Write the "READ" instruction for reading the second page of the bitstream data
8	……
9	Write the "PROGRAM" instruction for programming the nth page of the bitstream data
10	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
11	Write the "READ" instruction for reading the nth page of the bitstream data
12	Write the "WRDIS" instruction

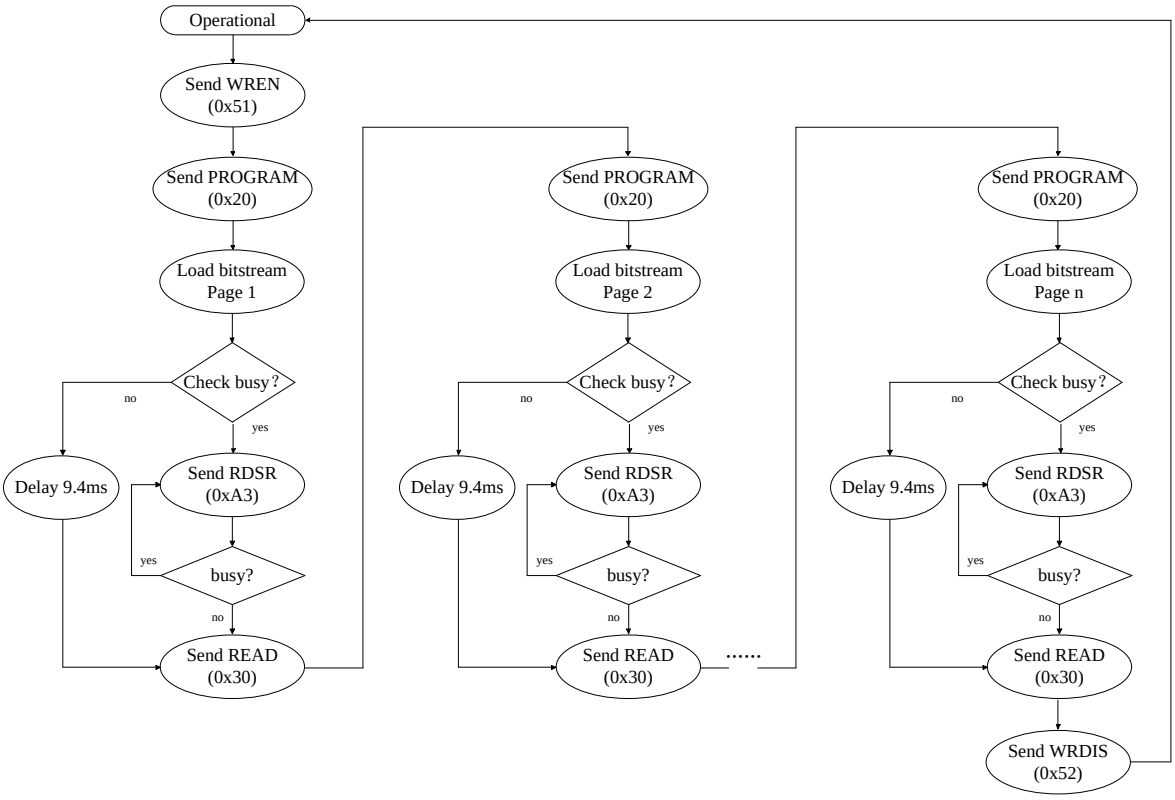

Figure 13-7 Program Bitstream

13.8 Erase Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-8 Erase Bitstream

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "ERASE_PAGE" instruction for erasing the first page of the bitstream data
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ" instruction for reading the first page of the bitstream data
5	Write the "ERASE_PAGE" instruction for erasing the second page of the bitstream data
6	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
7	Write the "READ" instruction for reading the second page of the bitstream data
8	……
9	Write the "ERASE_PAGE" instruction for erasing the nth page of the bitstream data
10	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
11	Write the "READ" instruction for reading the nth page of the bitstream data
12	Write the "WRDIS" instruction

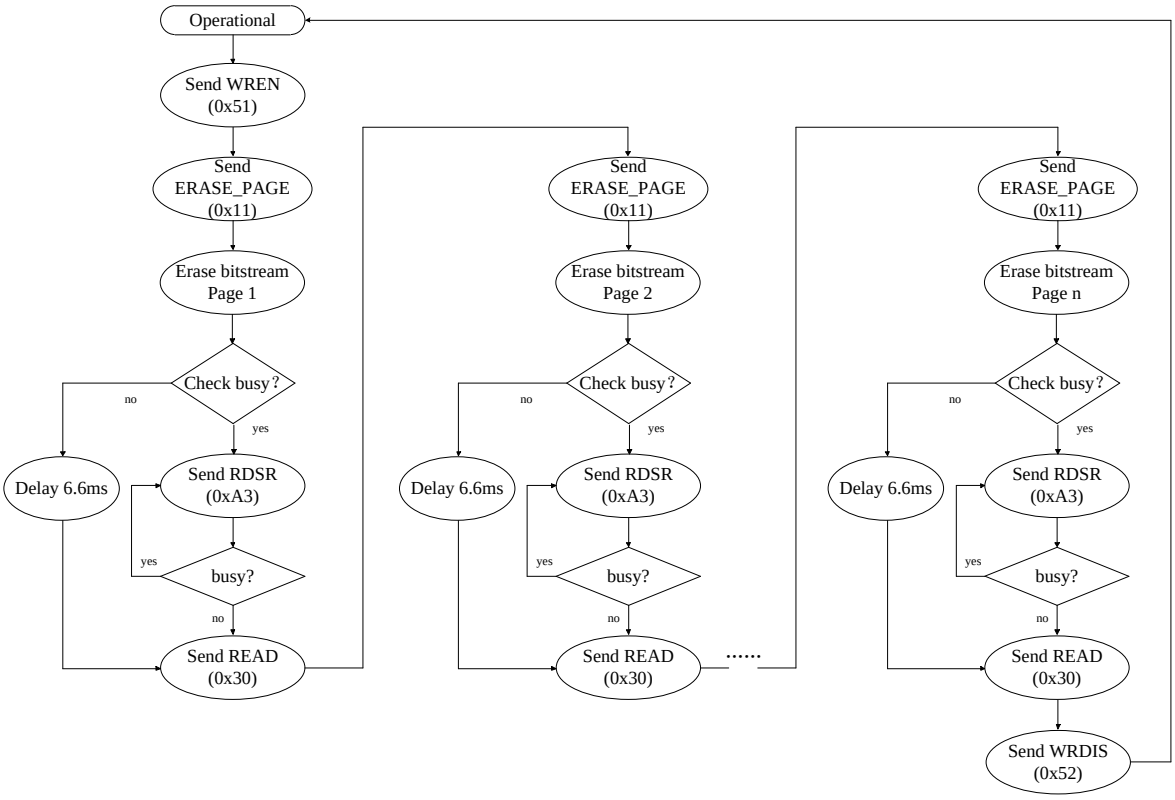

Figure 13-8 Erase Bitstream

13.9 Read Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-9 Read Bitstream

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "READ" instruction for reading bitstream
3	Write the "WRDIS" instruction

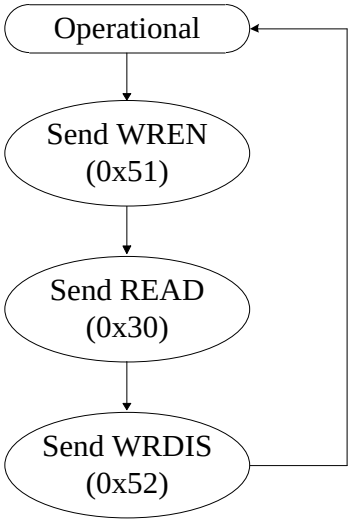

Figure 13-9 Read Bitstream

13.10 Program User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-10 Program User Flash

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "PROGRAM" instruction for programming the first page of the user Flash data
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ" instruction for reading the first page of the user Flash data
5	Write the "PROGRAM" instruction for programming the second page of the user Flash data
6	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
7	Write the "READ" instruction for reading the second page of the user Flash data
8	……
9	Write the "PROGRAM" instruction for programming the nth page of the user Flash data
10	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
11	Write the "READ" instruction for reading the nth page of the user Flash data
12	Write the "WRDIS" instruction

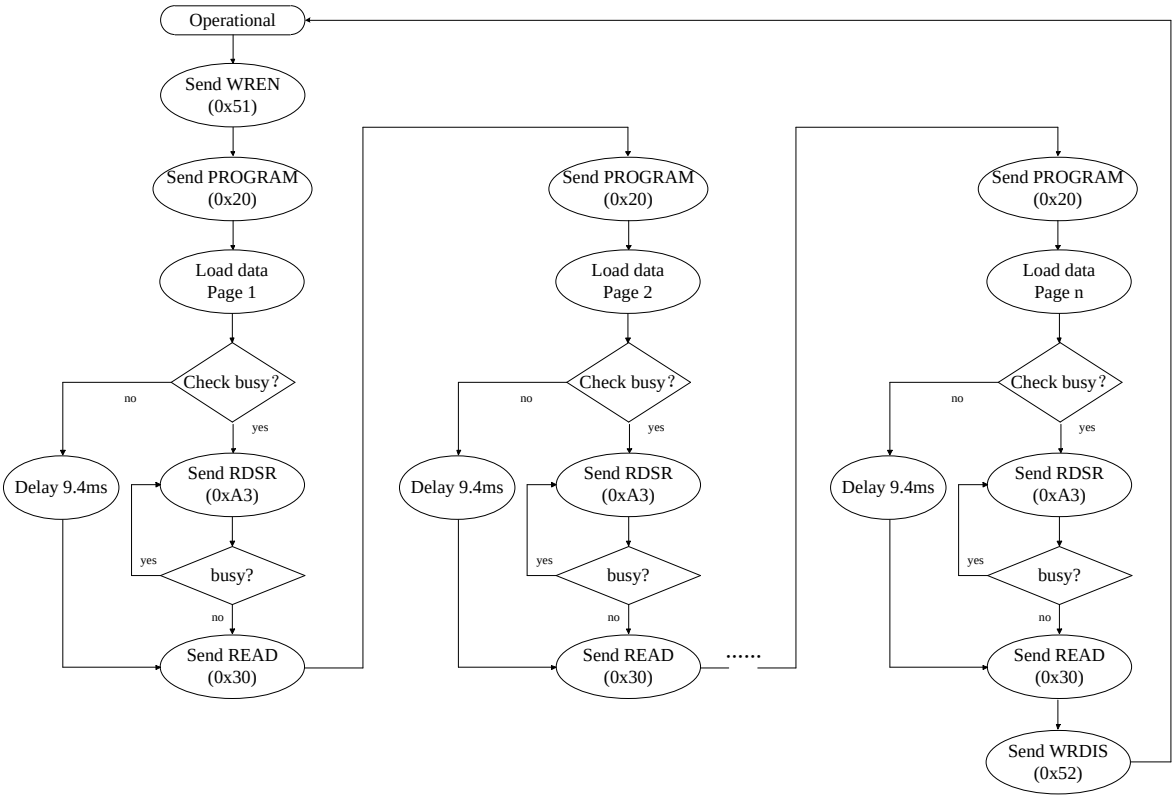

Figure 13-10 Program User Flash

13.11 Erase User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-11 Erase User Flash

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "ERASE_PAGE" instruction for erasing the first page of the user Flash data
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ" instruction for reading the first page of the user Flash data
5	Write the "ERASE_PAGE" instruction for erasing the second page of the user Flash data
6	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
7	Write the "READ" instruction for reading the second page of the user Flash data
8	……
9	Write the "ERASE_PAGE" instruction for erasing the nth page of the user Flash data
10	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
11	Write the "READ" instruction for reading the nth page of the user Flash data
12	Write the "WRDIS" instruction

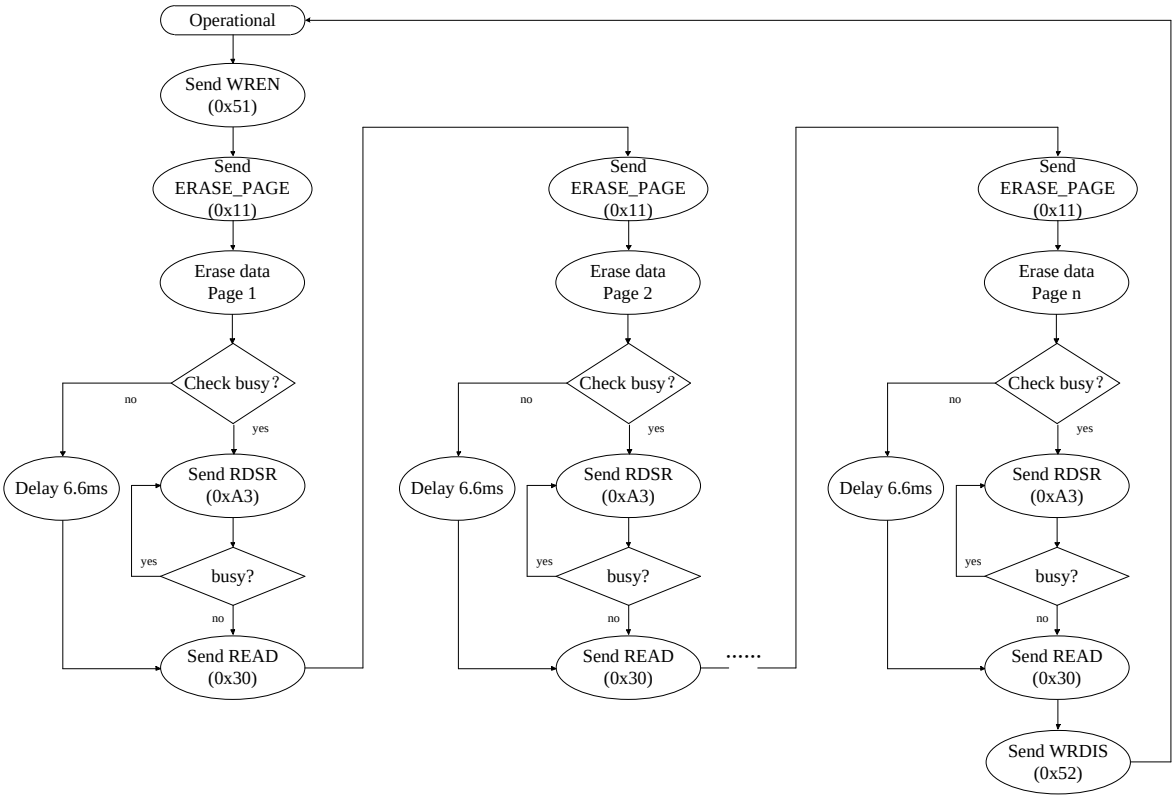

Figure 13-11 Erase User Flash

13.12 Erase Bitstream and User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-12 Erase Bitstream and User Flash

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "ERASE" instruction
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "READ" instruction for reading the data of the bitstream and user Flash
5	Write the "WRDIS" instruction

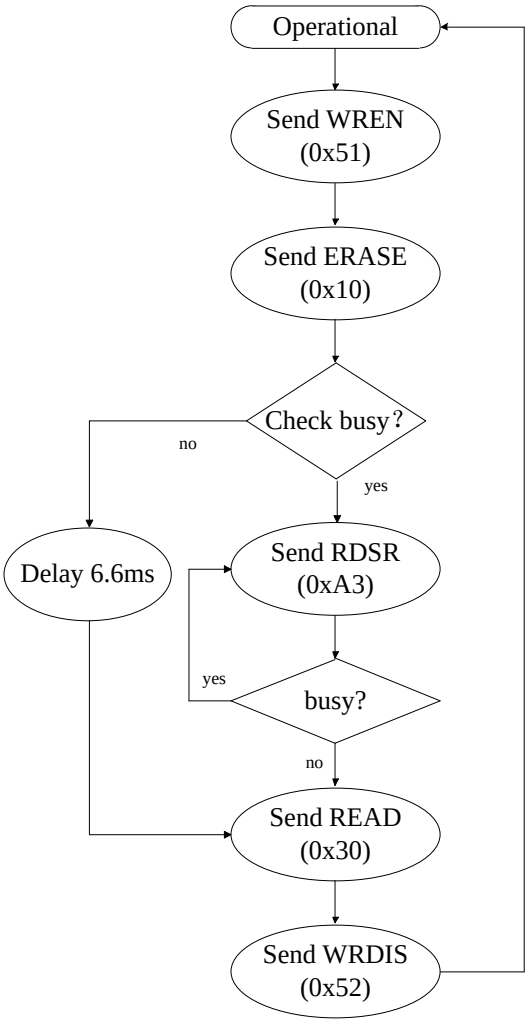

Figure 13-12 Erase Bitstream and User Flash

13.13 Read User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-13 Read User Flash

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "READ" instruction for reading the data of user Flash
3	Write the "WRDIS" instruction

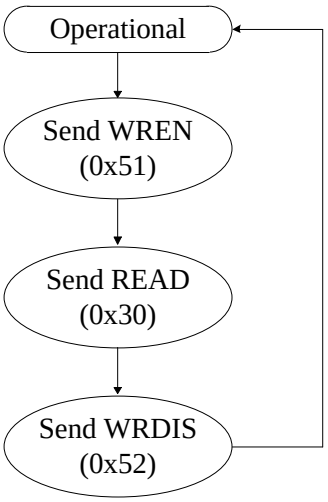

Figure 13-13 Read User Flash

13.14 Lock embedded Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

Table 13-14 Lock embedded Flash

Steps	Operation Description
1	Write the "WREN" instruction
2	Write the "PROGRAM_LOCK" instruction
3	Write the "RDSR" instruction and read the STATUSR register in a loop until the busy is detected as low
4	Write the "RDSR" instruction
5	Write the "WRDIS" instruction

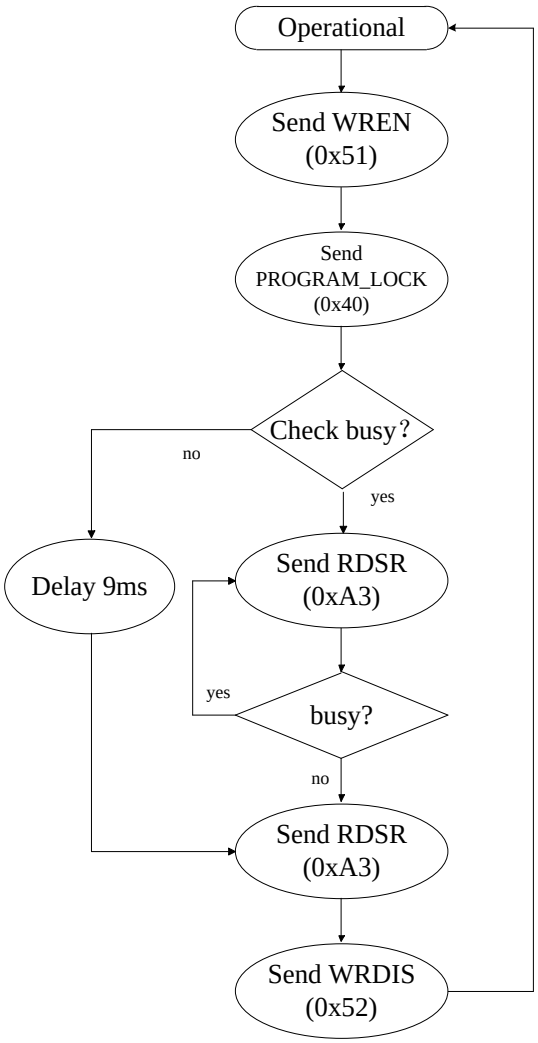

Figure 13-14 Lock embedded Flash

13.15 ISC Upgrade

Ensure the embedded Flash is in the wakeup state. After the "ISC_ENABLE" instruction is loaded, the device first samples the output value of the pins to the boundary scan register, and then hands over control of the output pins to the boundary scan register (only supported by PGC10K)

Table 13-15 ISC upgrade

Steps	Operation Description
1	Load the "ISC_ENABLE" instruction
2	If using the SPI interface, wait for 100 SCK clock cycles (100 SCK clocks are generated when chip select is inactive); If using the I^2C interface, wait for 1ms
3	Load the "RESET" instruction
4	Load the "RESET" instruction, cycle out the value of the status register until the wakeup_over bit is detected as 1

Steps	Operation Description
5	Load the "ISC_DISABLE" instruction
6	User logic functions normally

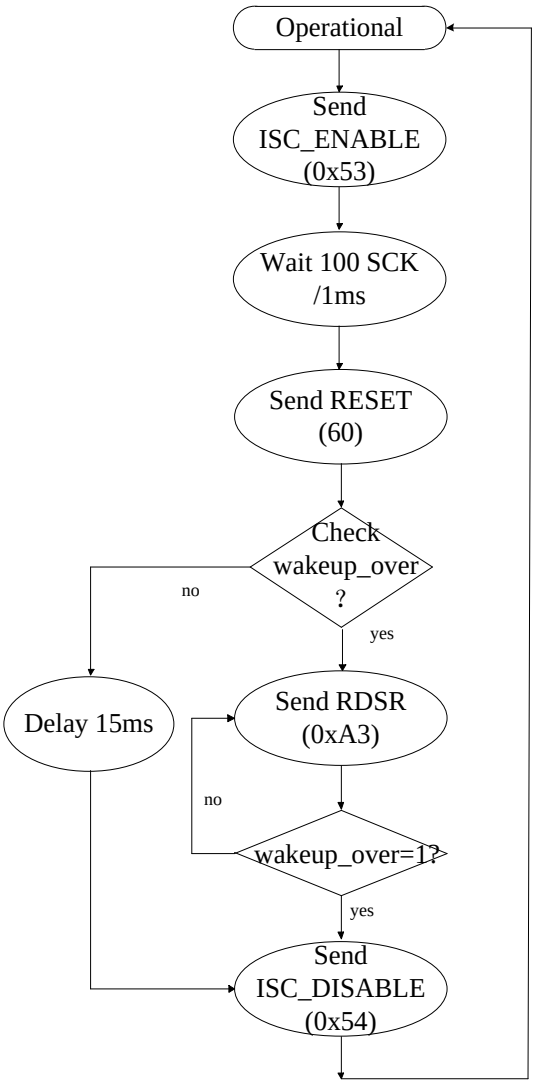

Figure 13-15 ISC upgrade

Chapter 14 Appendix 4

This section primarily describes the instructions for the slave SPI interface. When writing an instruction, write the most significant bit of the instruction first; when writing an address, write the most significant bit of the address first. During instruction operations, FCSI_N must be kept low. Before or after FCSI_N is pulled low, it must maintain a high level for 8 clock cycles. During this period, the clock must toggle normally. For detailed timing, refer to Figure 14-1 - Figure 14-27.

14.1 NOP

No Operation instruction

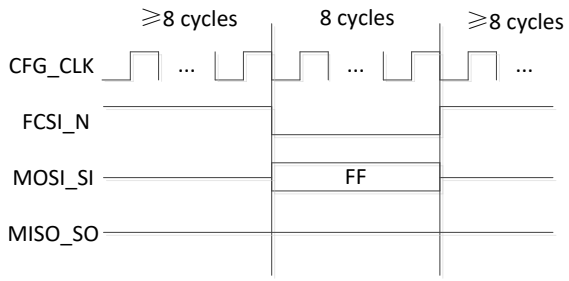

Figure 14-1 NOP

14.2 RDID

"Read IDCODE" instruction.

Used to read back the IDCODE of PGC devices, starting with the least significant bit

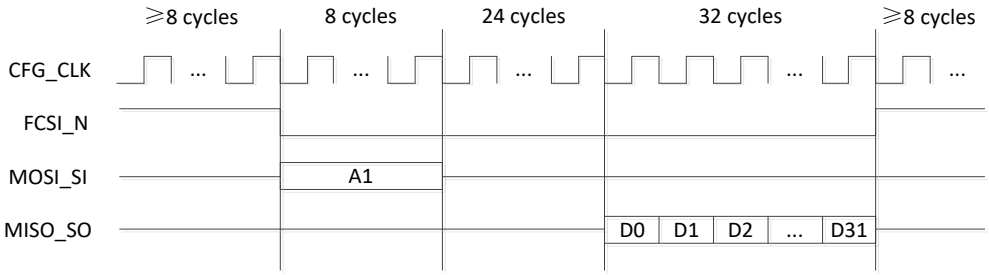

Figure 14-2 RDID

14.3 RDSR

"Read Status Register" instruction.

Used to read back the status register, starting with the least significant bit.

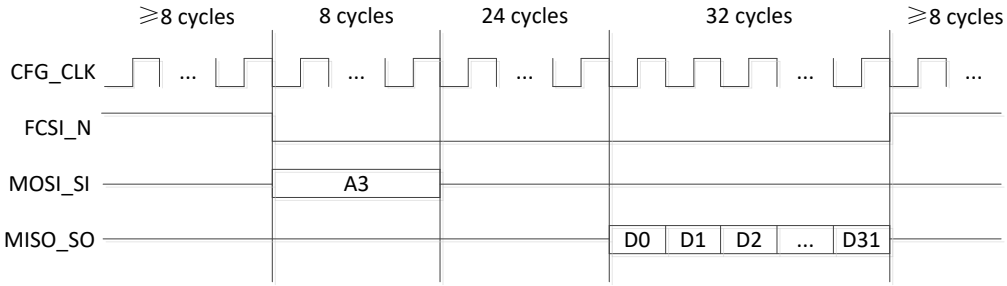

Figure 14-3 RDSR

Continuously read the status register

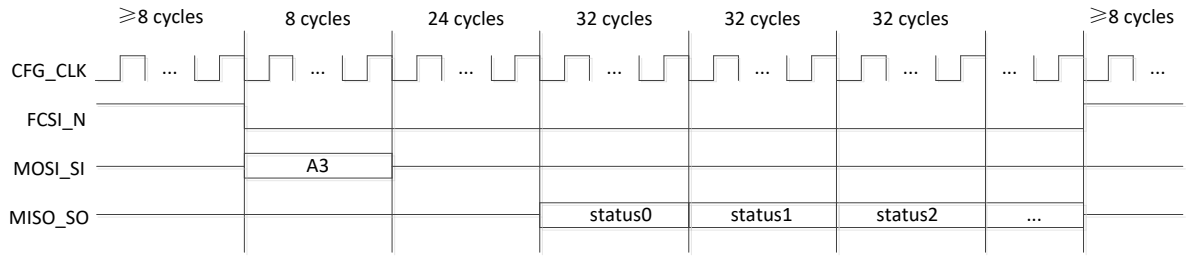

Figure 14-4 RDSR

14.4 RDUSER

"Read USERCODE" instruction.

Used to read back the USERCODE of PGC devices, starting with the least significant bit.

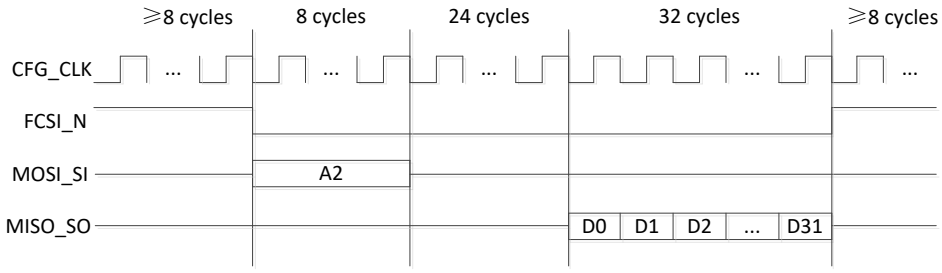

Figure 14-5 RDUSER

14.5 CFG

"Configuration" instruction.

Used to load bitstream and configure PGC devices.

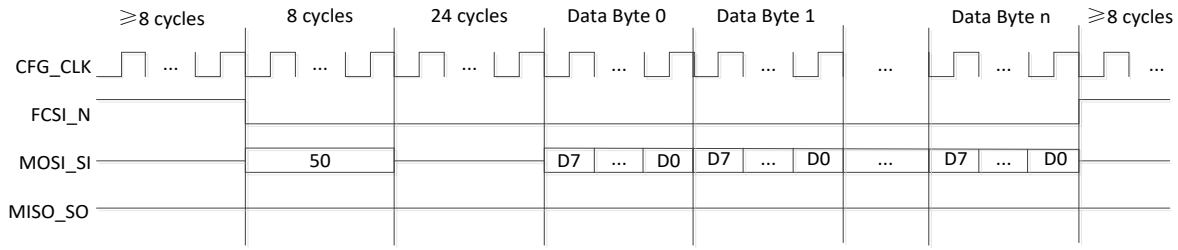

Figure 14-6 CFG

14.6 RDUID

"Read UID" instruction.

Used to read back the UID of PGC devices, starting with the least significant bit.

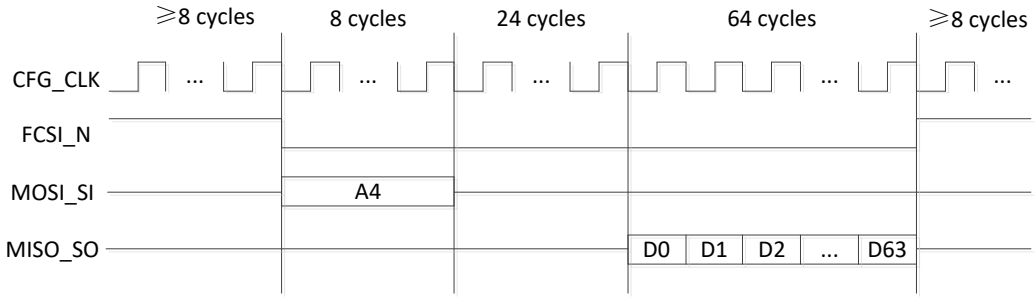

Figure 14-7 RDUID

Continuously read back the UID of PGC devices.

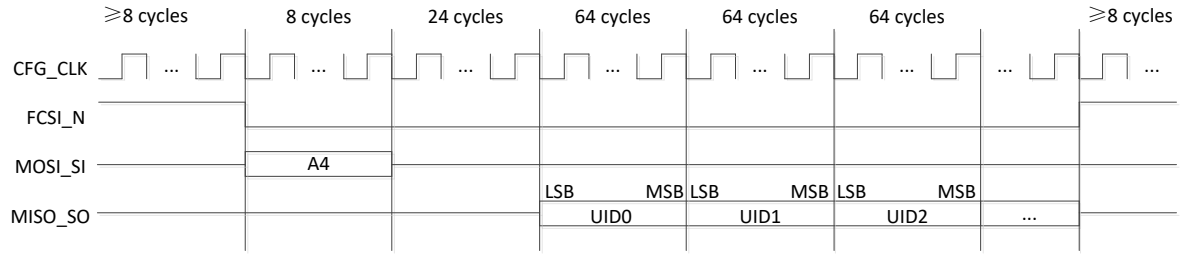

Figure 14-8 RDUID

14.7 RDLOCK

"Read Embedded Flash Lock Information" instruction.

Used to read back the bitstream tail page address and embedded Flash lock flag, starting with the least significant bit.

14.7.1 1K/2K/4K/7K

Sequentially shift out the address of the last bitstream page "lock_ra[0:8]", heap address "lock_ba[0:1]", and lock flag "lock" and "4'd0".

For 1K devices, "lock_ba[1:0]" is fixed at the value of "2'b11".

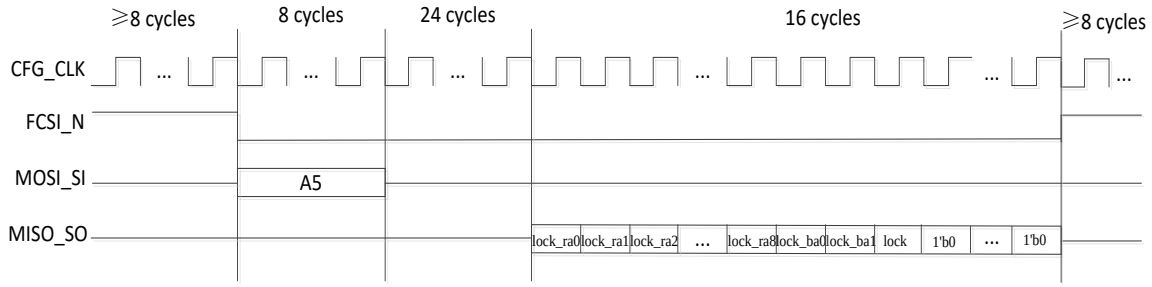

Figure 14-9 RDLOCK

For 2K devices, "lock_ba[1:0]" is the reserved heap address.

14.7.2 10K

Sequentially shift out the address of the last Master Self Download bitstream page "lock_ra[0: 9]", heap address "lock_ba[0:1]", and lock flag "lock" and "3'd0".

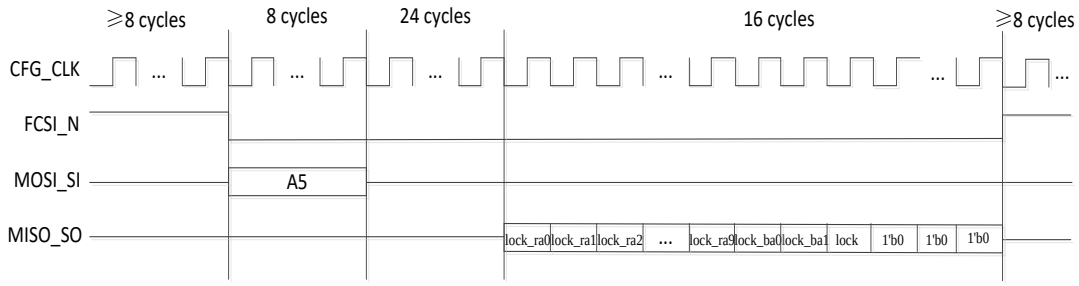

Figure 14-10 RDLOCK

Continuously read embedded Flash lock information.

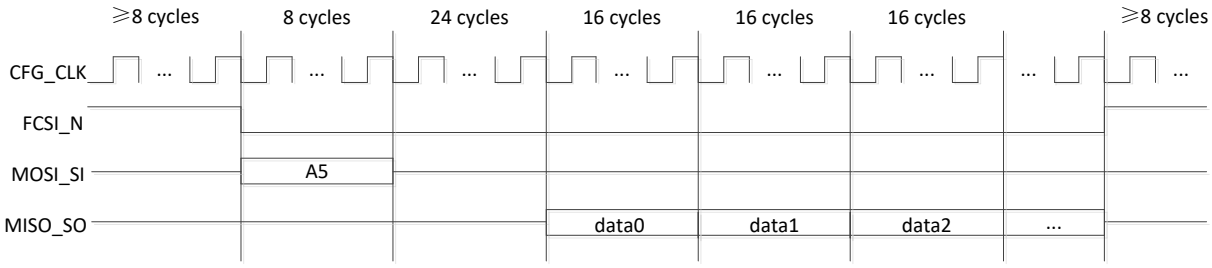

Figure 14-11 RDLOCK

14.8 WREN

"Write Enable" instruction.

Used before "Configuration" instructions and various "Program" instructions.

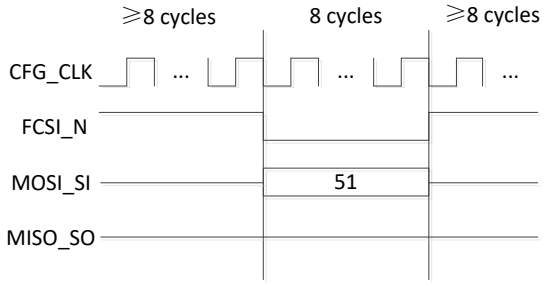

Figure 14-12 WREN

14.9 WRDIS

"Write Disable" instruction.

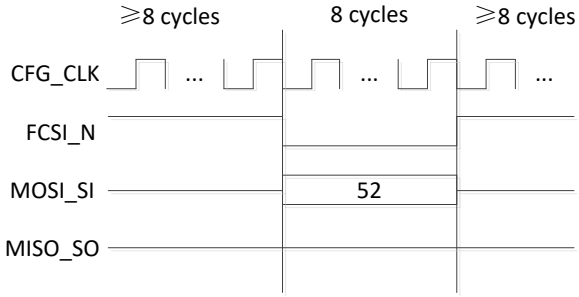

Figure 14-13 WRDIS

14.10 RESET

"Reset" instruction.

Reset the CCS parts except JTAG.

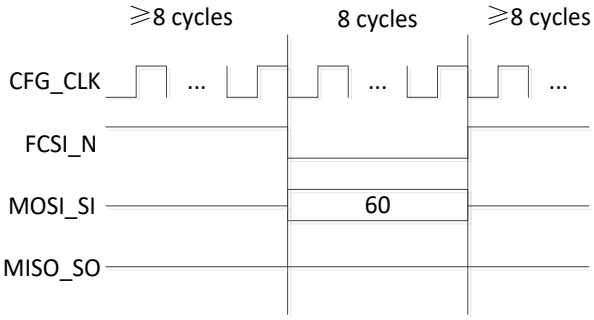

Figure 14-14 RESET

14.11 ERASE

"Erase" instruction.

Erase all normal pages of the embedded Flash and clears the lock flag of the embedded Flash.

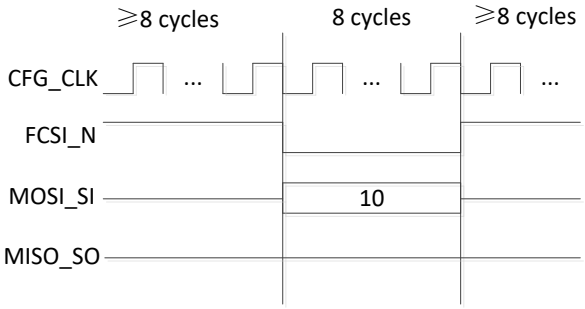

Figure 14-15 ERASE

14.12 ERASE_PAGE

"Page Erase" instruction.

Erase the normal pages of the embedded Flash.

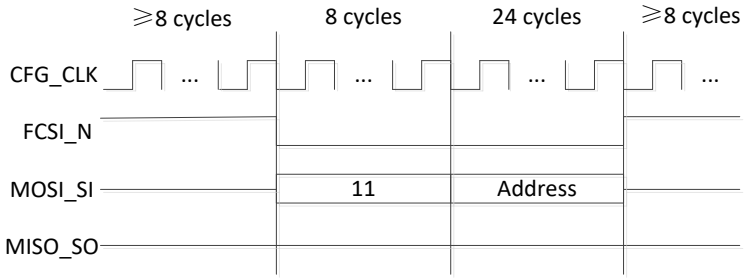

Figure 14-16 ERASE_PAGE

Table 14-1 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

14.13 ERASE_CTL

"Control Erase" instruction.

Used for erasing the feature control bits of embedded Flash.

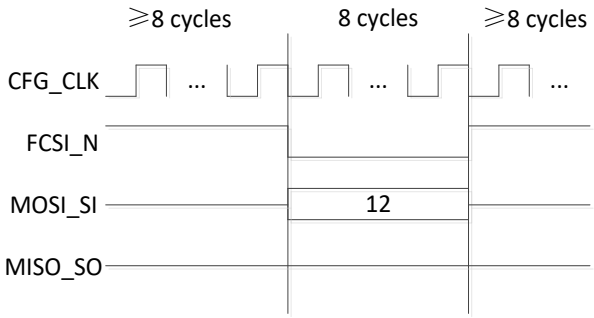

Figure 14-17 ERASE_CTL

14.14 PROGRAM

"Program" instruction.

Used for programming the normal pages of embedded Flash.

14.14.1 1K/2K/4K/7K

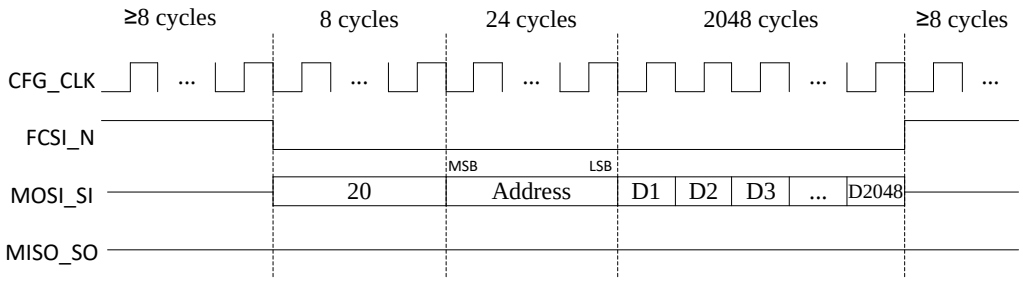

Figure 14-18 PROGRAM

Table 14-2 Address

Address	**Item**	**Description**
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

14.14.2 10K

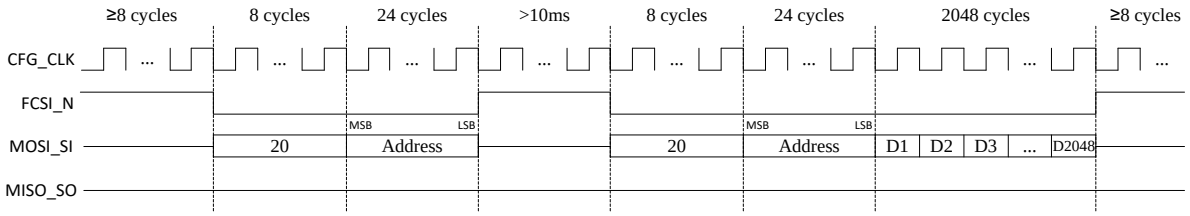

Figure 14-19 PROGRAM

Table 14-3 Address

Address	**Item**	**Description**
[23:20]	Reserved	Must be set to 4'b0000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[9:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

14.15 PROGRAM_CTL

"Control Program" instruction.

Used for programming the feature control bits of embedded Flash.

Data is sequentially moved into ctl[0:31].

The 24-bit address is reserved.

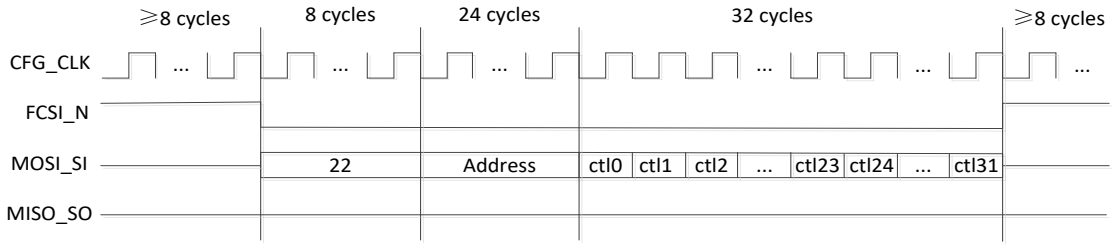

Figure 14-20 PROGRAM_CTL

14.16 READ

"Read" instruction.

Used for reading the normal pages of embedded Flash.

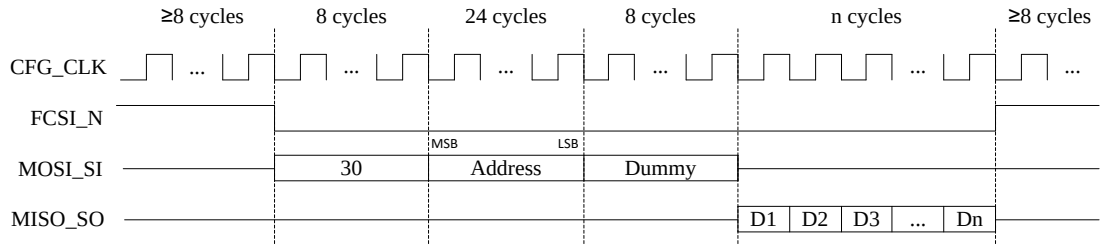

Figure 14-21 READ

14.16.1 1K/2K/4K/7K

Table 14-4 Address

Address	**Item**	**Description**
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

14.16.2 10K

Table 14-5 Address

Address	**Item**	**Description**
[23:20]	Reserved	Must be set to 4'b0000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[9:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

14.17 READ_CTL

"Control Read" instruction.

Used for reading the feature control bits of embedded Flash.

Data is sequentially moved out of ctl[0:31].

The 24-bit address is reserved.

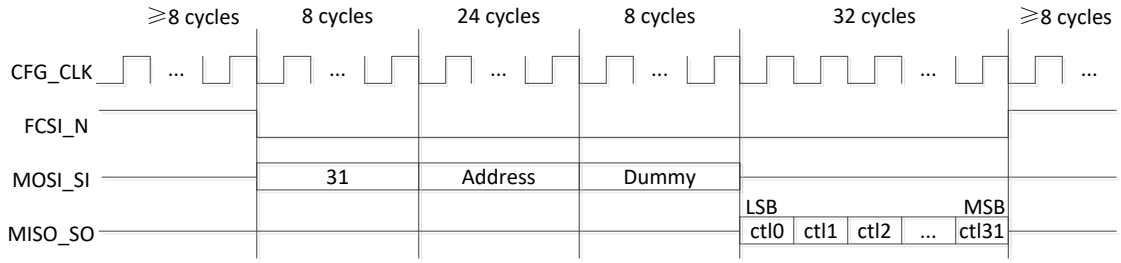

Figure 14-22 READ_CTL

14.18 PROGRAM_LOCK

"Lock" instruction.

Used for disabling program and read operations on embedded Flash.

Data sequentially shifts into the address of the last bitstream page "lock_ra[0:8]", heap address "lock_ba[0:1]", and lock flag "lock" and "4'd0".

For 1K devices, lock_ba[1:0] is the reserved bit.

For 2K devices, lock_ba[1:0] is the reserved heap address and should be set to 2'b11.

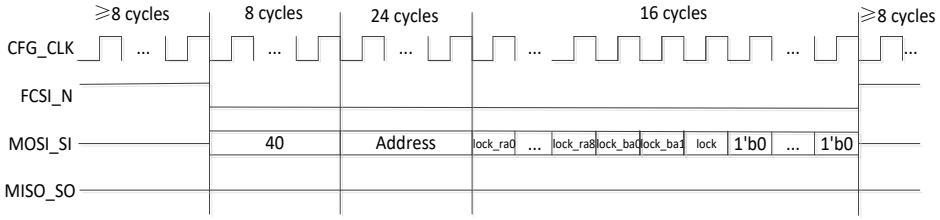

Figure 14-23 PROGRAM_LOCK

The 24-bit address is reserved.

14.19 EFlash_SLEEP

"Embedded Flash Sleep" instruction.

Used for putting embedded Flash into sleep mode.

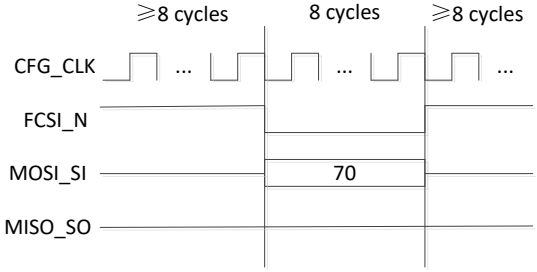

Figure 14-24 EFlash_SLEEP

14.20 EFlash_WAKEUP

"Embedded Flash Wake-up" instruction.

Used for waking up embedded Flash.

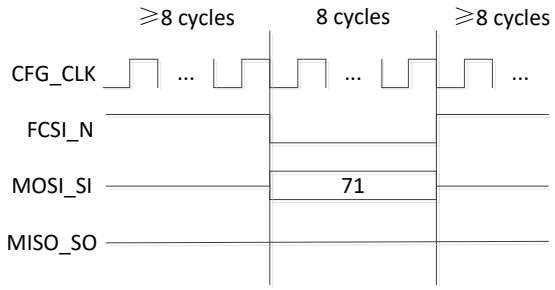

Figure 14-25 EFlash_WAKEUP

14.21 ISC_ENABLE

"ISC Enable" instruction.

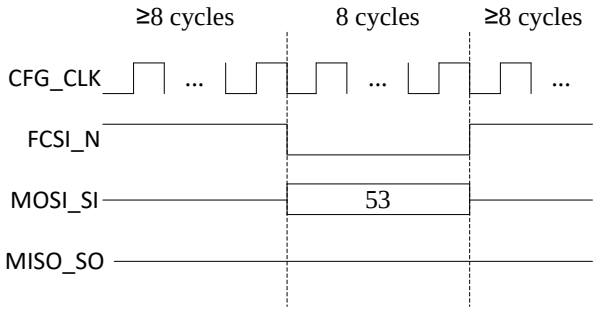

Figure 14-26 ISC_ENABLE

14.22 ISC_DISABLE

"ISC Disable" instruction.

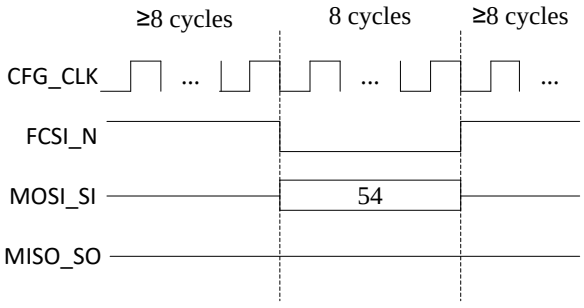

Figure 14-27 ISC_DISABLE

Chapter 15 Appendix 5

I^{2}C Interface Instruction Timing Description

15.1 NOP

No Operation instruction

15.1.1 7-Bit Addressing

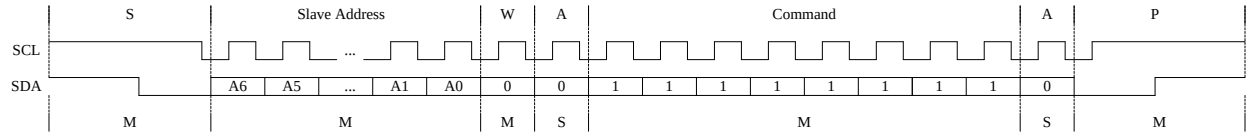

Figure 15-1 NOP

15.1.2 10-bit addressing

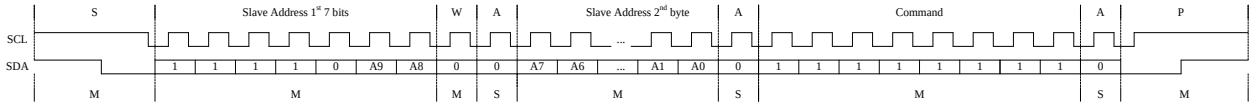

Figure 15-2 NOP

15.2 RDID

"Read IDCODE" instruction.

Used to read back the IDCODE of CPLD devices, starting with the least significant bit.

15.2.1 7-Bit Addressing

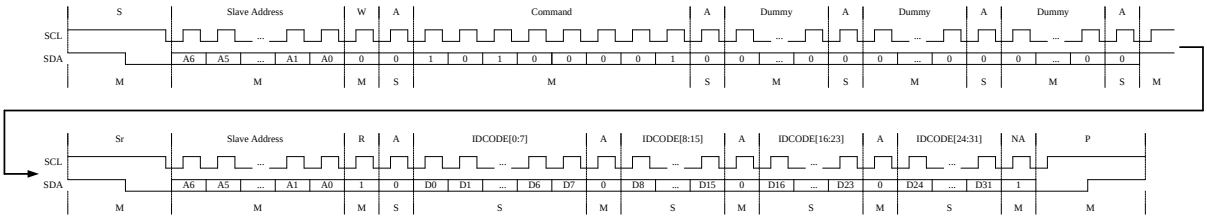

Figure 15-3 RDID

15.2.2 10-bit addressing

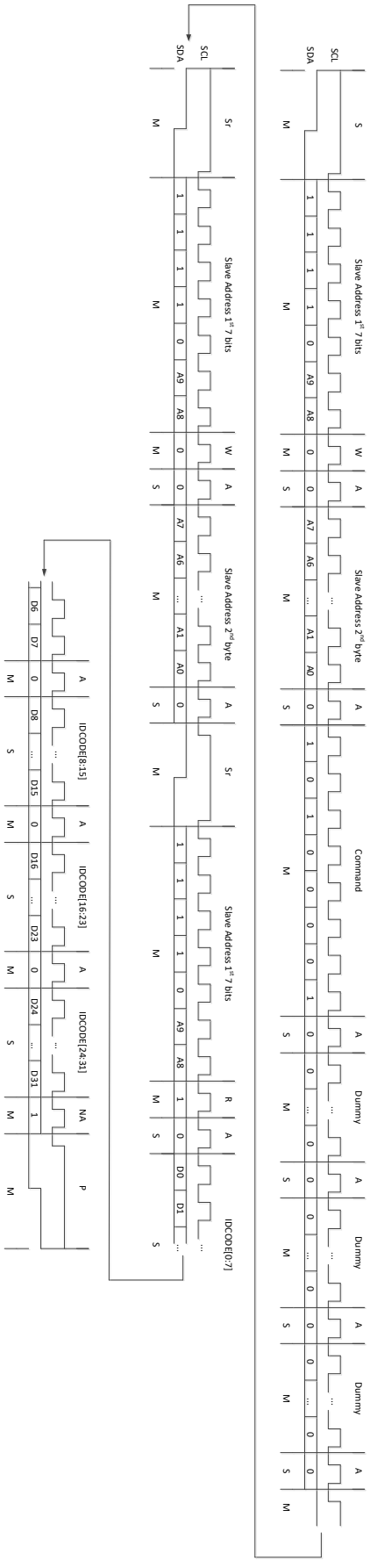

Figure 15-4 RDID

15.3 RDUSER

"Read USERCODE" instruction.

Used to read back the USERCODE of CPLD devices, starting with the least significant bit.

15.3.1 7-Bit Addressing

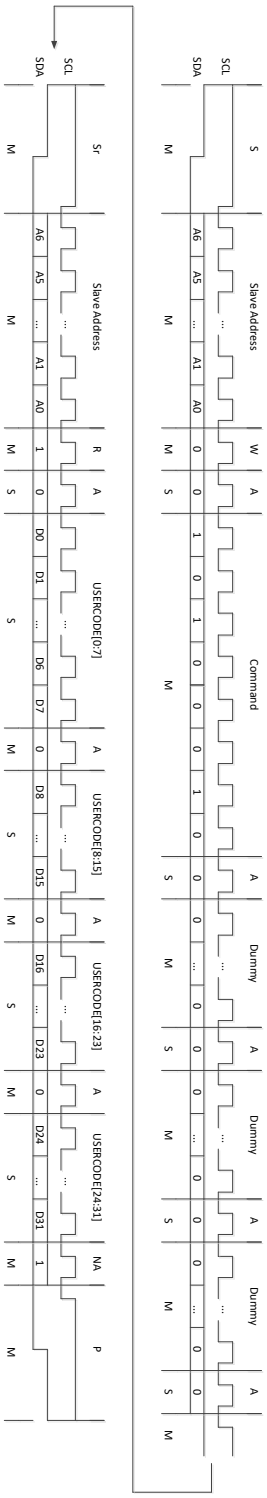

Figure 15-5 RDUSER

15.3.2 10-bit addressing

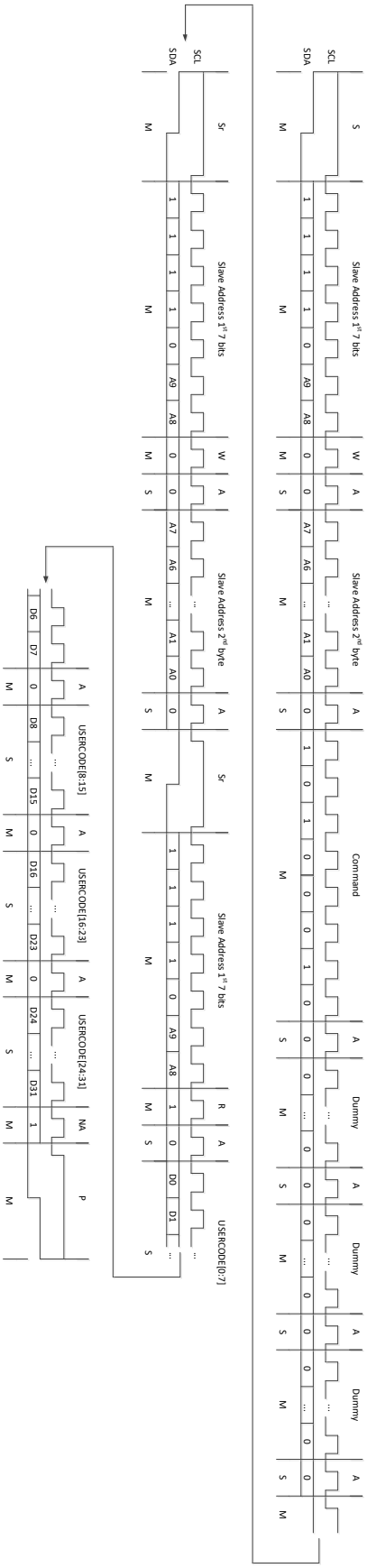

Figure 15-6 RDUSER

15.4 RDSR

"Read Status Register" instruction.

Used to read back the status register, starting with the least significant bit.

15.4.1 7-Bit Addressing

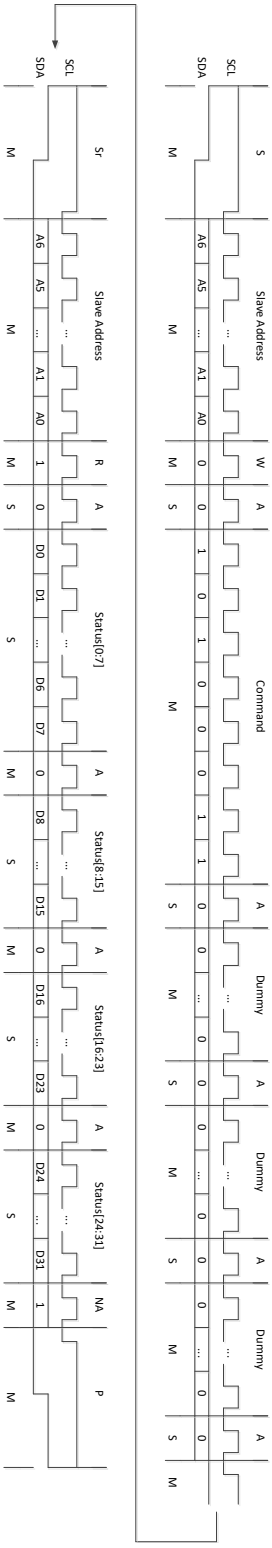

Figure 15-7 RDSR

Continuously read the status register

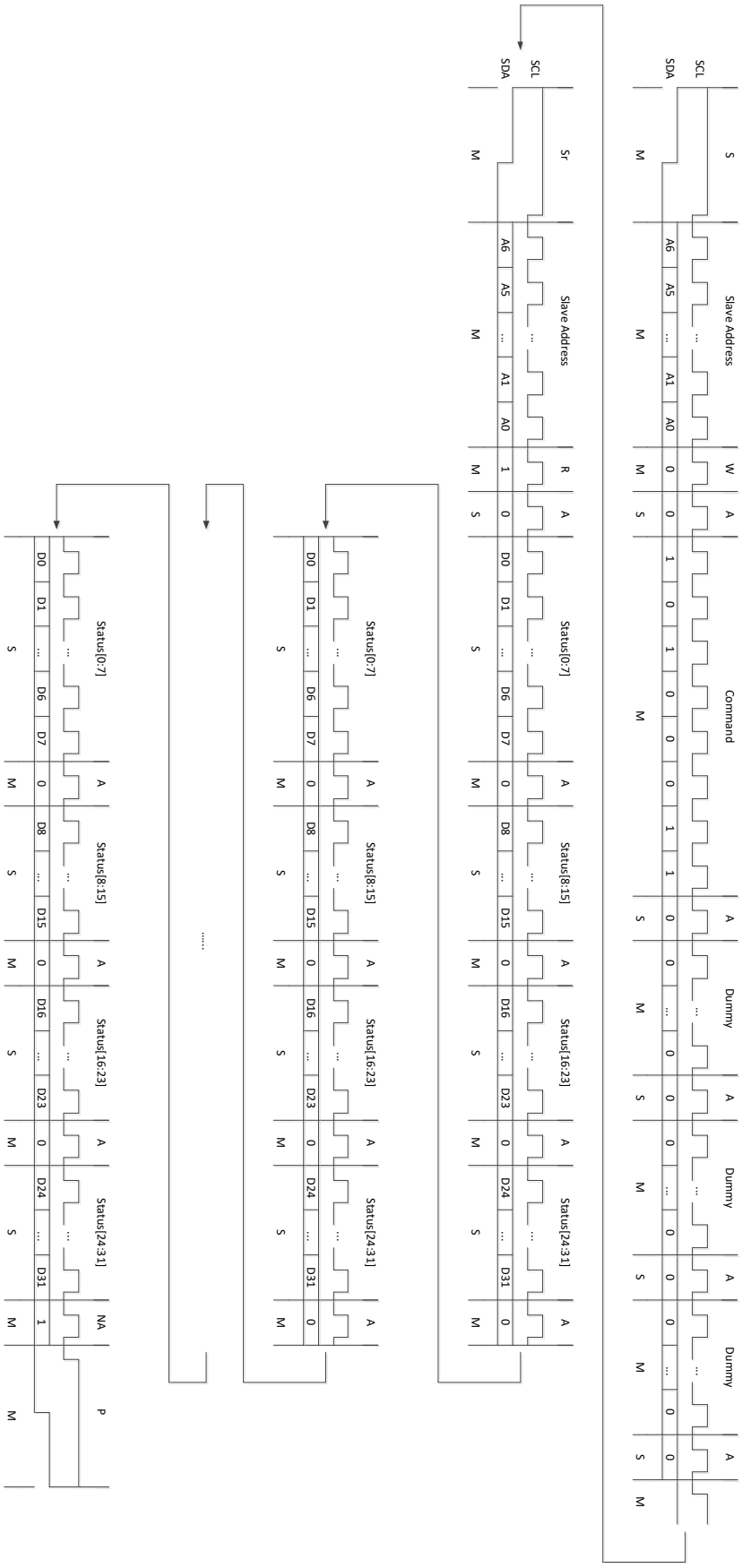

Figure 15-8 RDSR

15.4.2 10-bit addressing

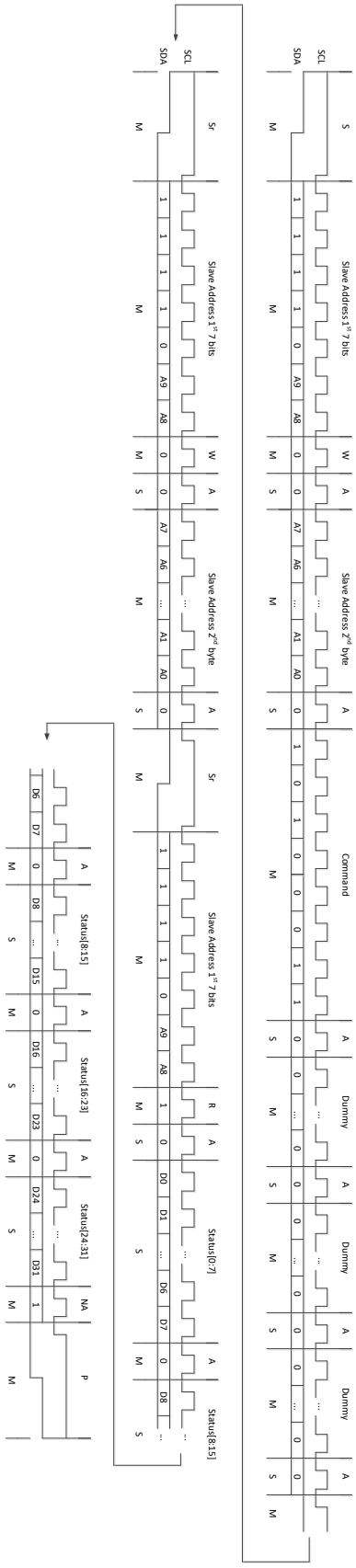

Figure 15-9 RDSR

Continuously read the status register

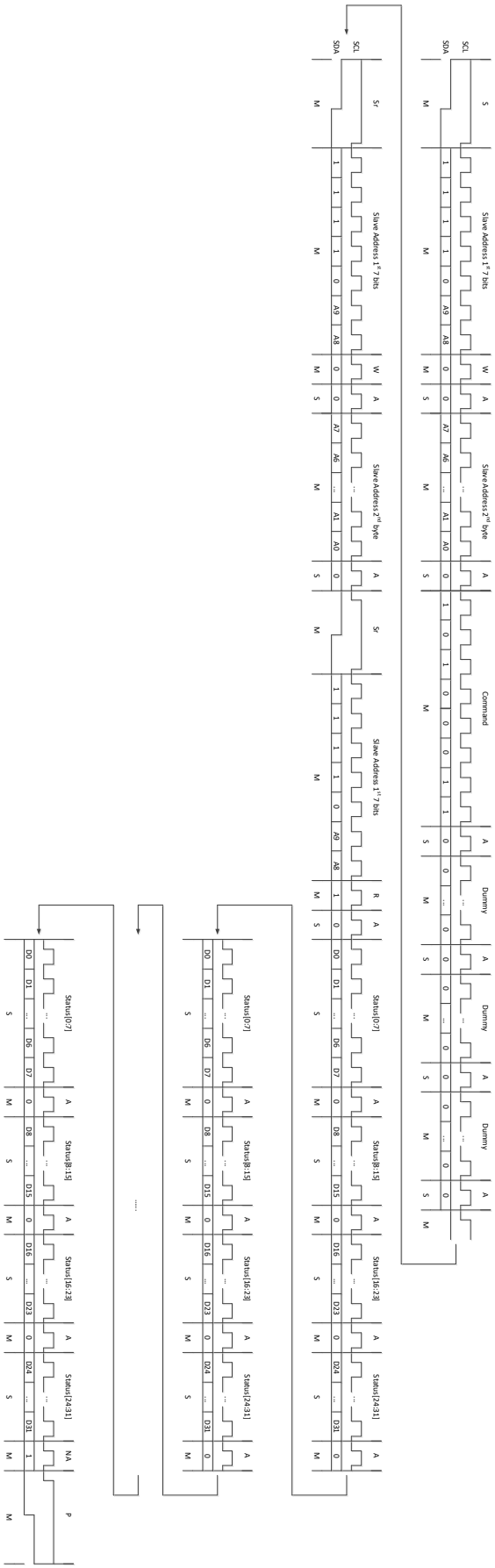

Figure 15-10 RDSR

15.5 RDUID

"Read UID" instruction.

Read back the UID of CPLD devices, starting with the least significant bit.

15.5.1 7-Bit Addressing

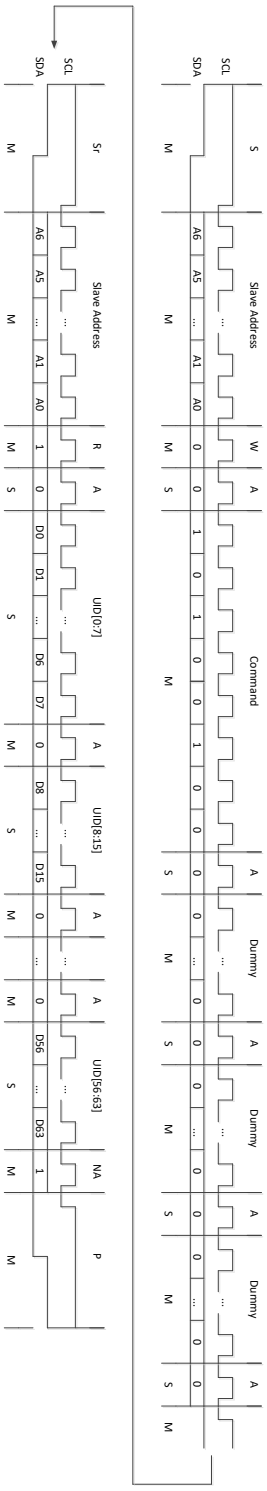

Figure 15-11 RDUID

Read back continuously the UID of CPLD devices.

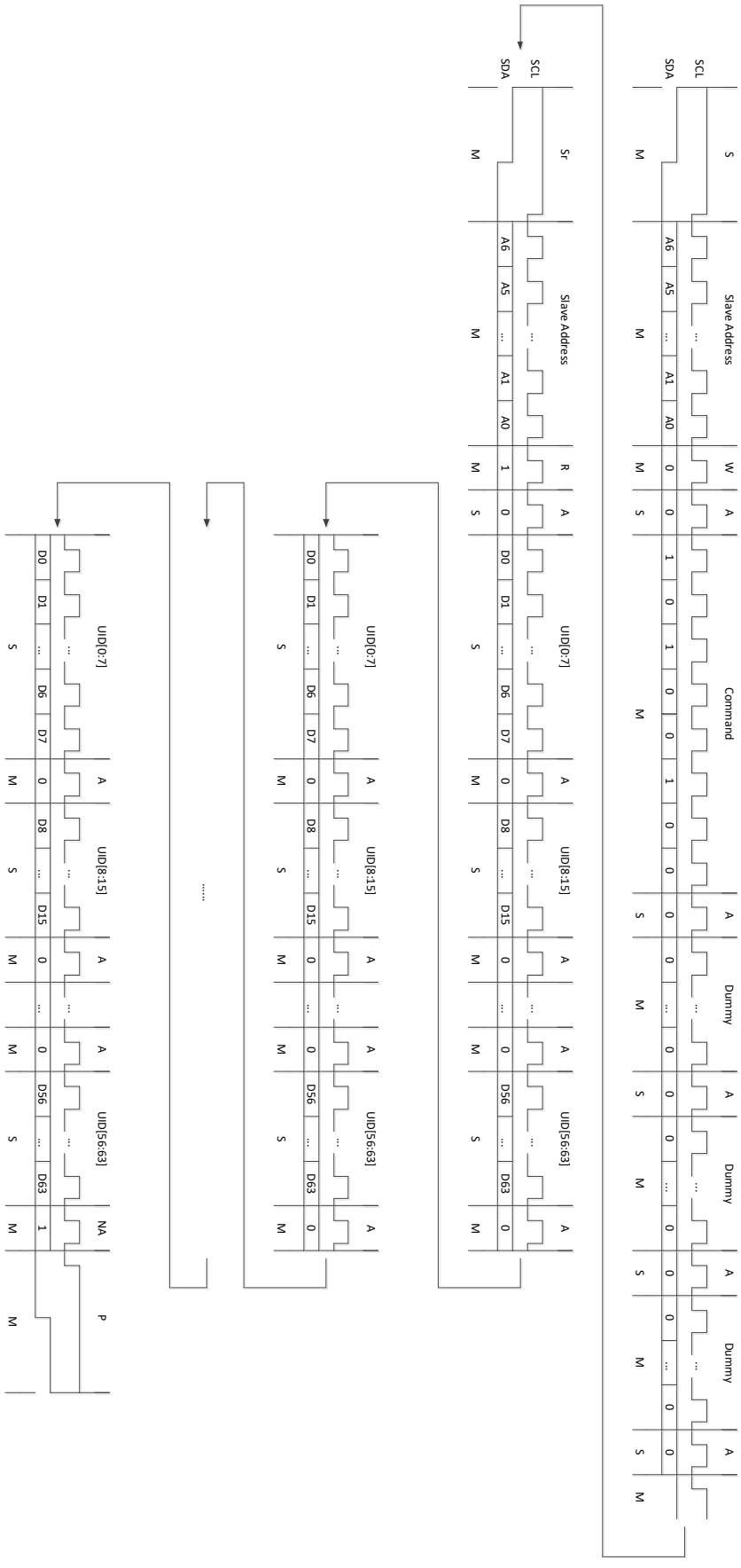

Figure 15-12 RDUID

15.5.2 10-bit addressing

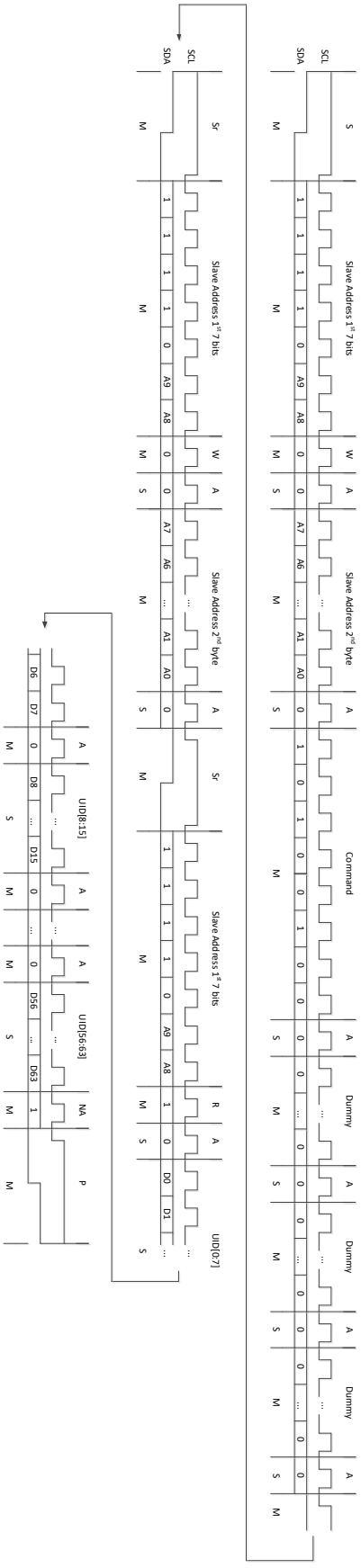

Figure 15-13 RDUID

Read back continuously the UID of CPLD devices.

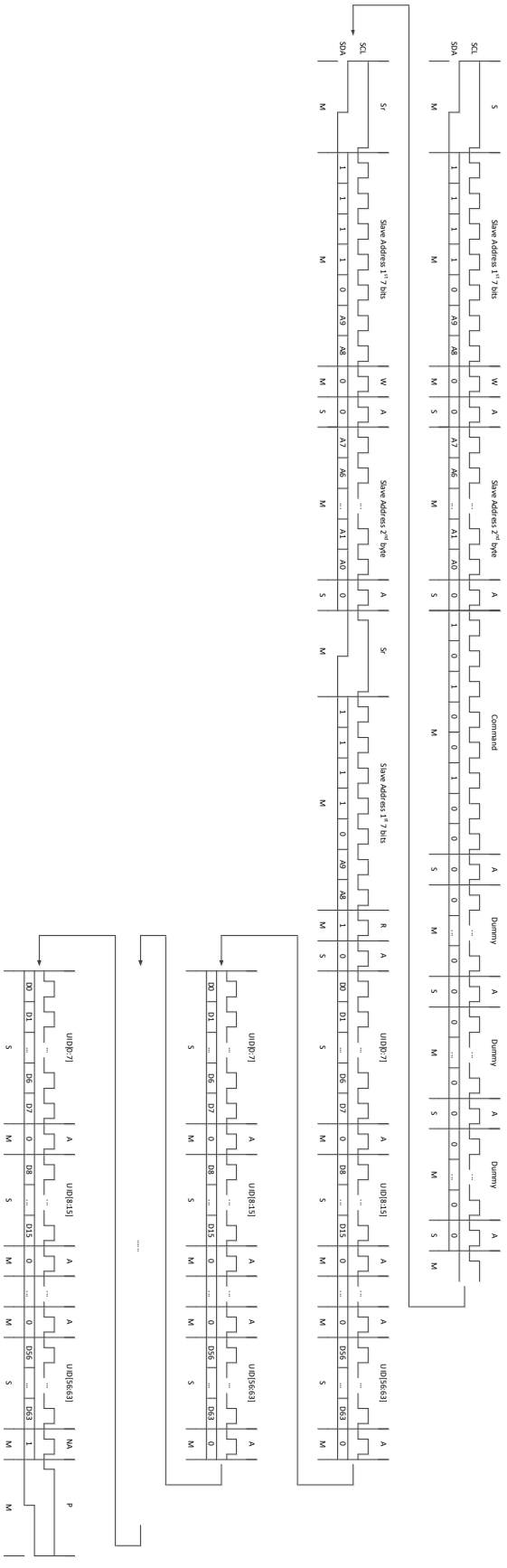

Figure 15-14 RDUID

15.6 RDLOCK

"Read Embedded Flash Lock Information" instruction.

Used to read back the bitstream tail page address and embedded Flash lock flag, starting with the least significant bit. Sequentially shift out the address of the last bitstream page "lock_ra[0:8]", heap address "lock_ba[0:1]", and lock flag "lock" and "4'd0".

For 1K devices, "lock_ba[1:0]" is fixed at the value of "2'b11".

For 2K devices, "lock_ba[1:0]" is the reserved heap address.

15.6.1 7-Bit Addressing

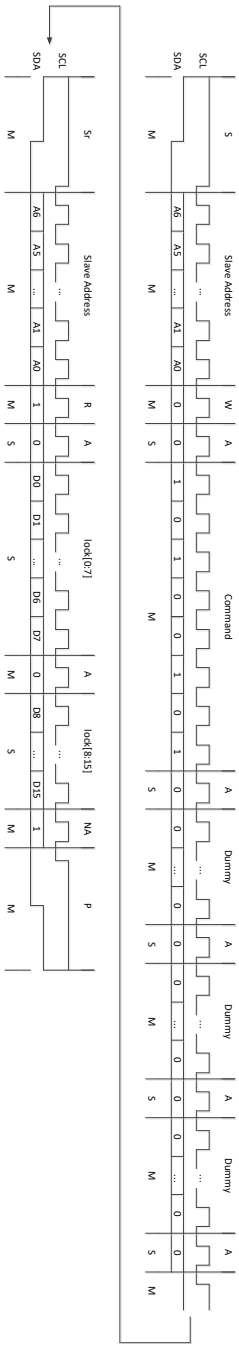

Figure 15-15 RDLOCK

Continuously read embedded Flash lock information.

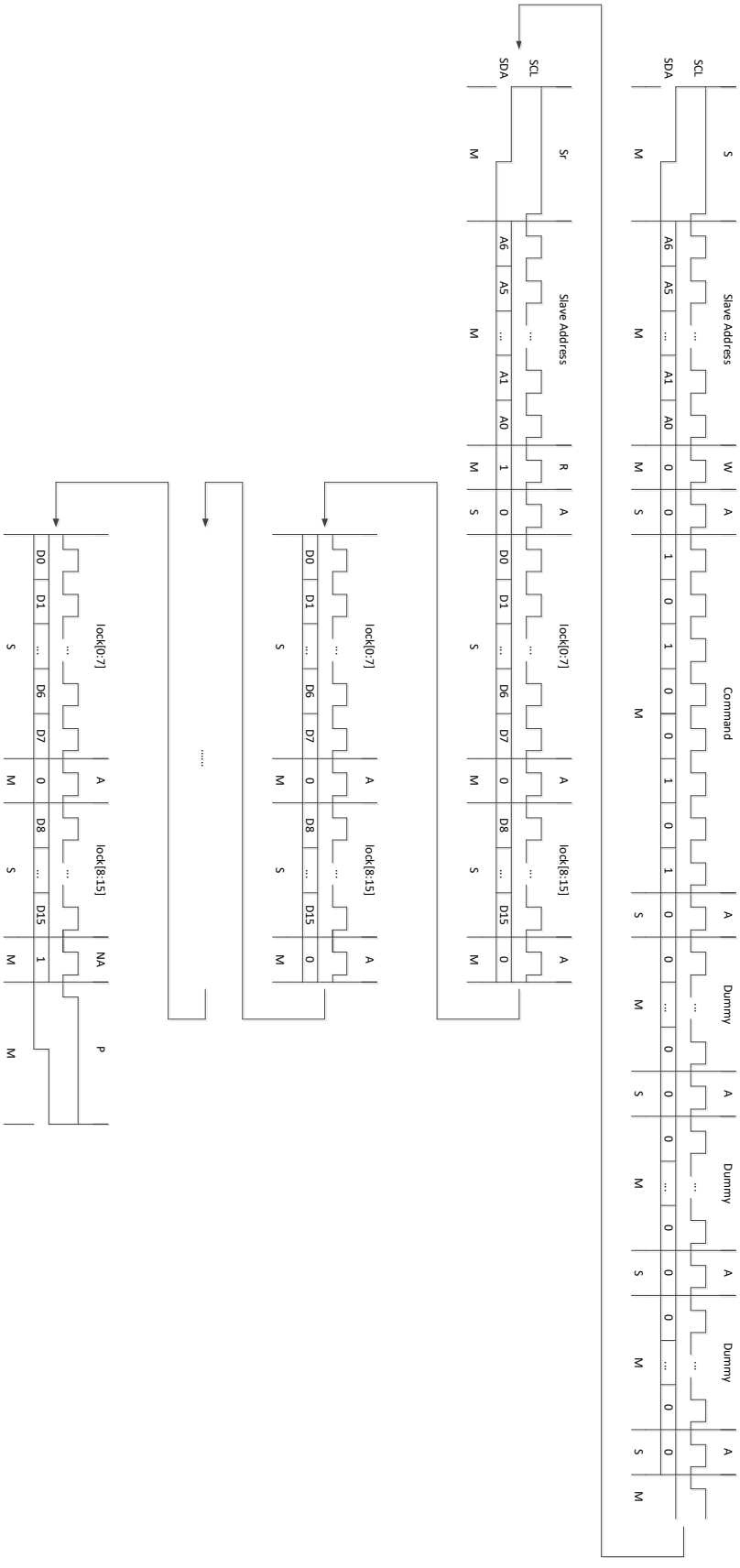

Figure 15-16 RDLOCK

15.6.2 10-bit addressing

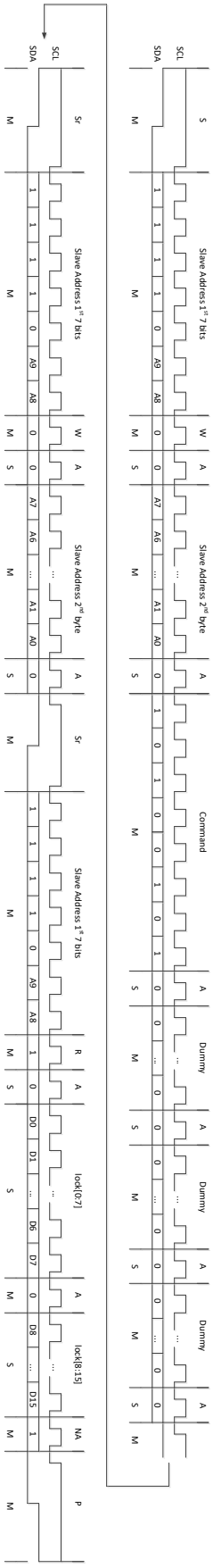

Figure 15-17 RDLOCK

Continuously read embedded Flash lock information.

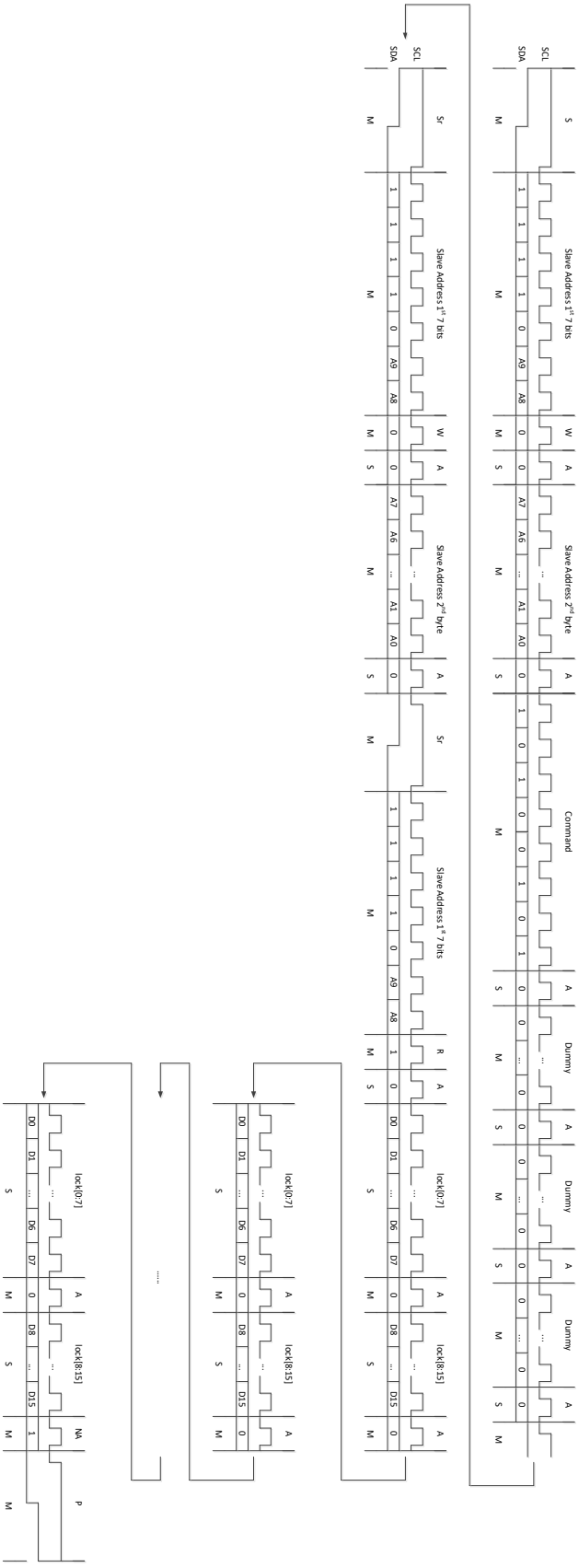

Figure 15-18 RDLOCK

15.7 CFG

"Configuration" instruction.

Used for loading the bitstream and configuring CPLD devices. Write the higher bits of each byte first.

15.7.1 7-Bit Addressing

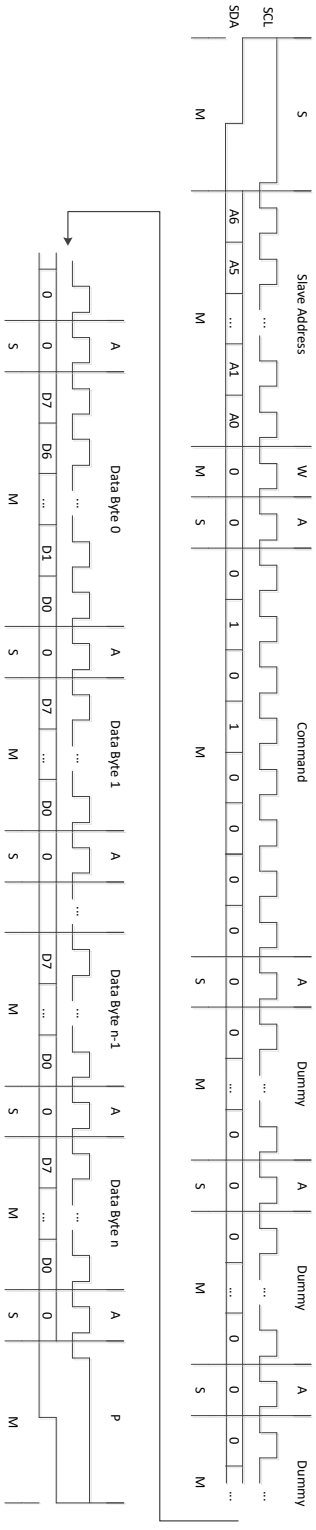

Figure 15-19 CFG

15.7.2 10-bit addressing

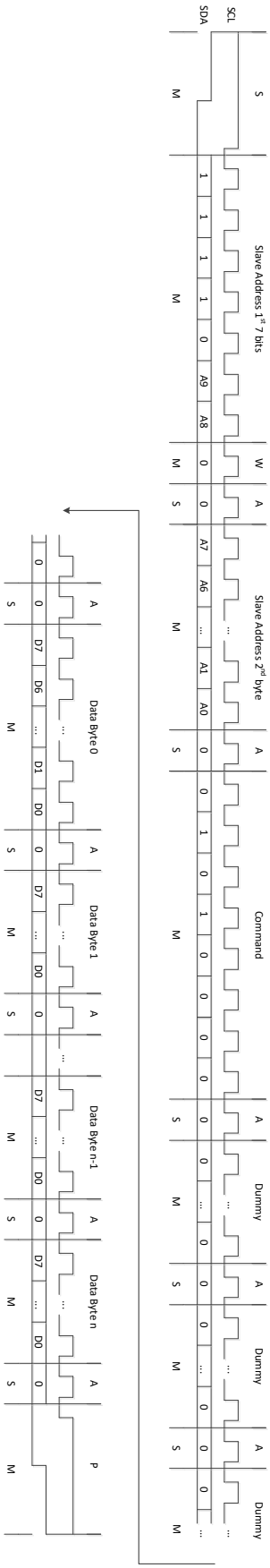

Figure 15-20 CFG

15.8 WREN

"Write Enable" instruction.

Used before "Configuration" instructions and various "Program" instructions.

15.8.1 7-Bit Addressing

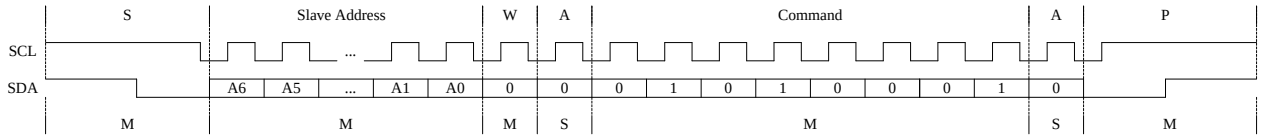

Figure 15-21 WREN

15.8.2 10-bit addressing

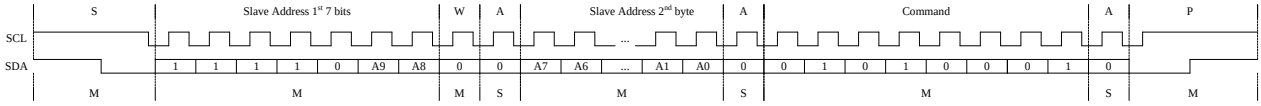

Figure 15-22 WREN

15.9 WRDIS

"Write Disable" instruction.

15.9.1 7-Bit Addressing

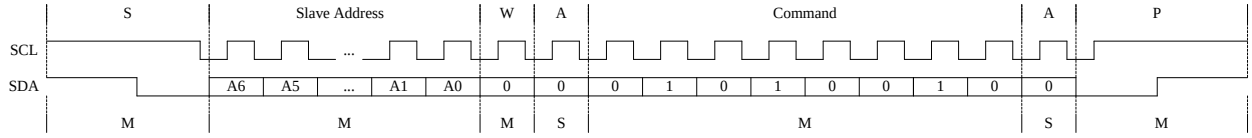

Figure 15-23 WRDIS

15.9.2 10-bit addressing

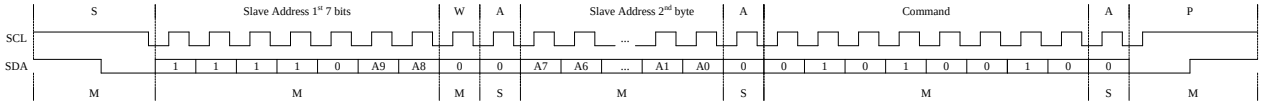

Figure 15-24 WRDIS

15.10 RESET

"Reset" instruction.

Reset the CCS parts except JTAG.

15.10.1 7-Bit Addressing

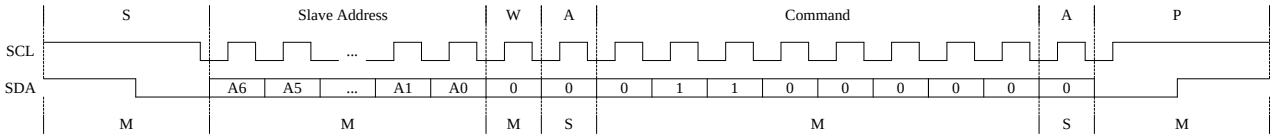

Figure 15-25 RESET

15.10.2 10-bit addressing

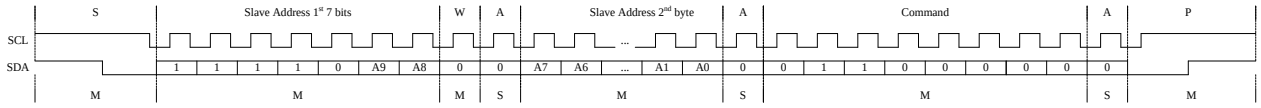

Figure 15-26 RESET

15.11 ERASE

"Erase" instruction.

Erase all normal pages of the embedded Flash and clears the lock flag of the embedded Flash.

15.11.1 7-Bit Addressing

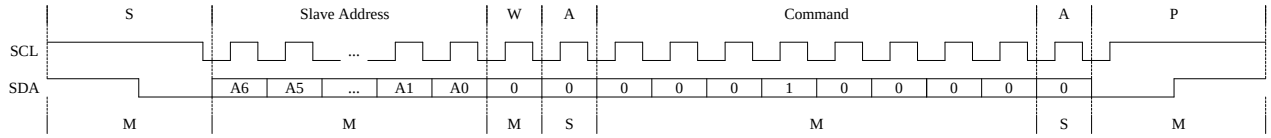

Figure 15-27 ERASE

15.11.2 10-bit addressing

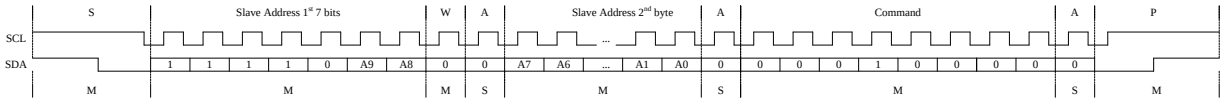

Figure 15-28 ERASE

15.12 ERASE_PAGE

"Page Erase" instruction.

Erase the normal pages of the embedded Flash.

15.12.1 7-Bit Addressing

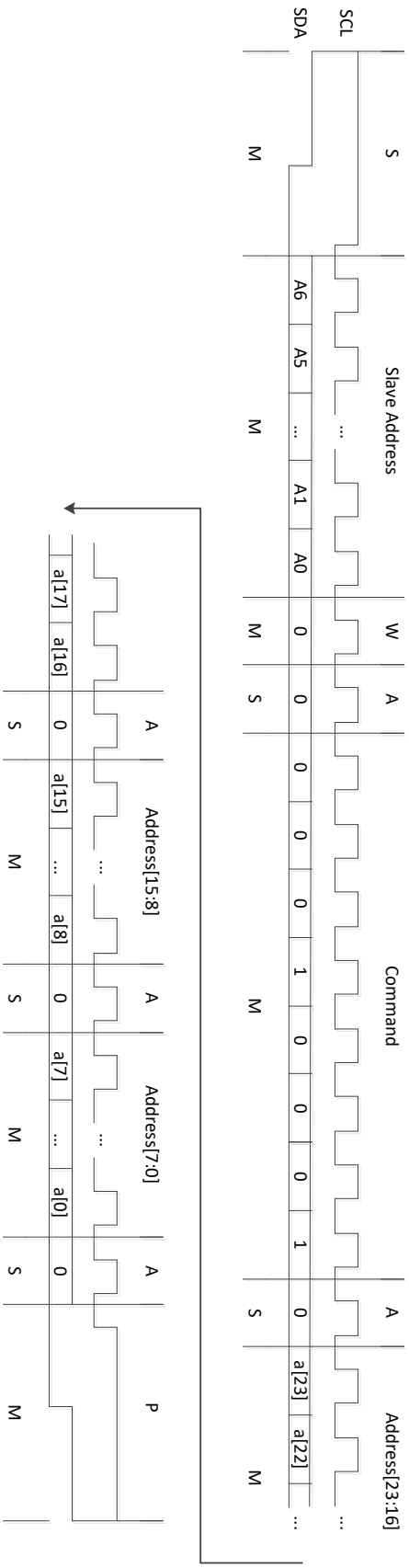

Figure 15-29 ERASE_PAGE

Table 15-1 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

15.12.2 10-bit addressing

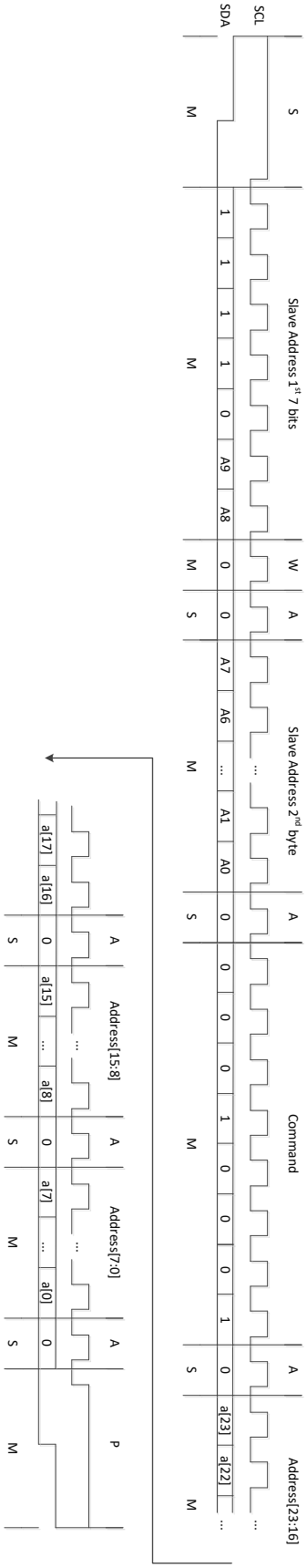

Figure 15-30 ERASE_PAGE

Table 15-2 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

15.13 ERASE_CTL

"Control Erase" instruction.

Used for erasing the feature control page of embedded Flash.

15.13.1 7-Bit Addressing

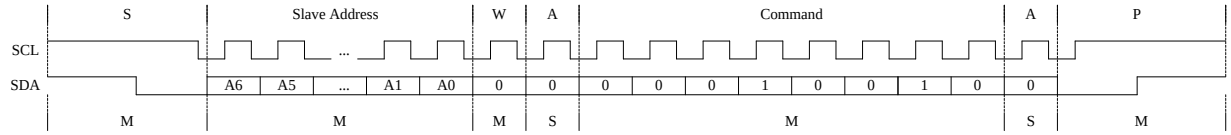

Figure 15-31 ERASE_CTL

15.13.2 10-bit addressing

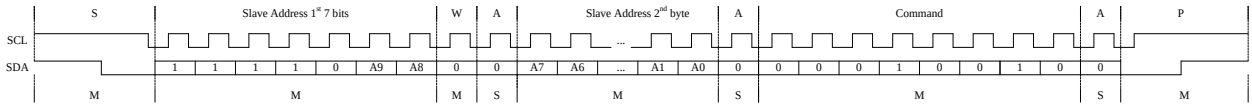

Figure 15-32 ERASE_CTL

15.14 PROGRAM

"Program" instruction.

Used for programming the normal pages of embedded Flash.

15.14.1 7-Bit Addressing

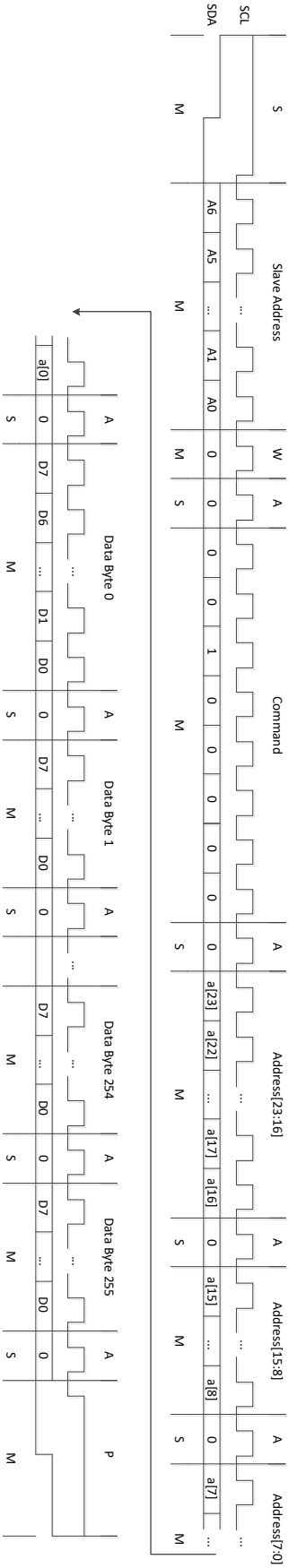

Figure 15-33 PROGRAM

15.14.1.1 1K/2K/4K/7K

Table 15-3 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

15.14.1.2 10K

Table 15-4 Address

Address	Item	Description
[23:20]	Reserved	Must be set to 4'b0000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[9:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

15.14.2 10-bit addressing

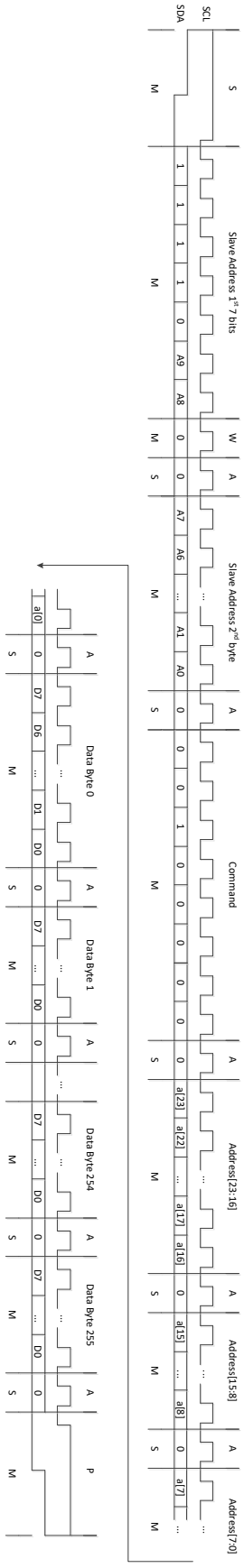

Figure 15-34 PROGRAM

15.14.2.1 1K/2K/4K/7K

Table 15-5 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

15.14.2.2 10K

Table 15-6 Address

Address	Item	Description
[23:20]	Reserved	Must be set to 4'b0000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[9:0]	Page Address
[7:0]	Reserved	Needs to be set to 8'b00000000

15.15 PROGRAM_CTL

"Control Program" instruction.

Used for programming the feature control bits.

15.15.1 7-Bit Addressing

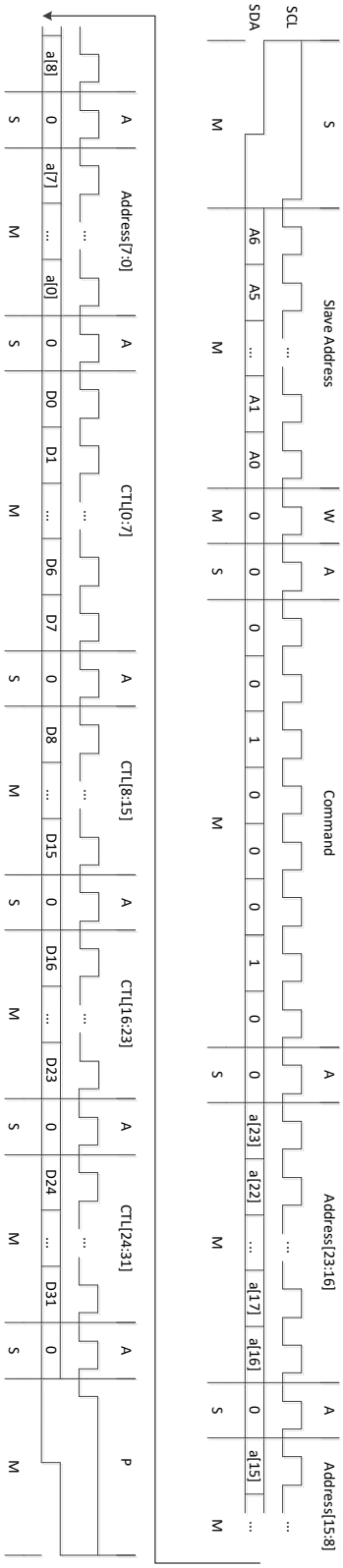

Figure 15-35 PROGRAM_CTL

Table 15-7 Address

Address	Item	Description

Address	Item	Description
[23:0]	Reserved	Needs to be set to 24'd0

15.15.2 10-bit addressing

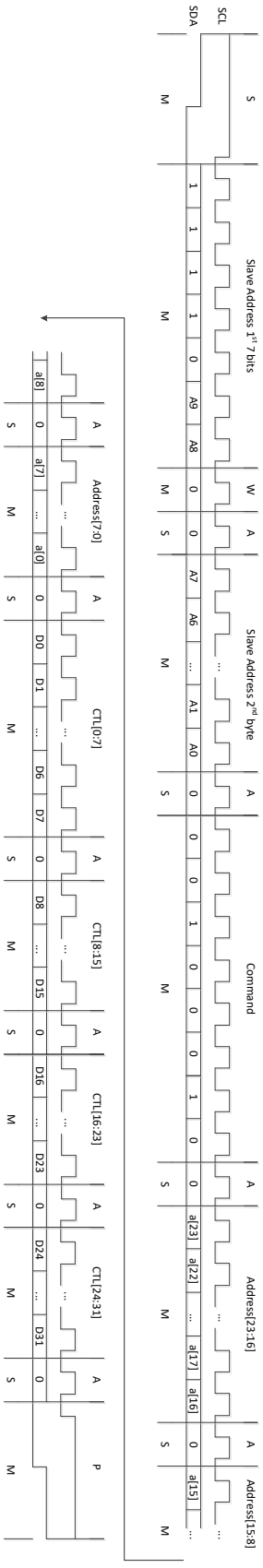

Figure 15-36 PROGRAM_CTL

Table 15-8 Address

Address	Item	Description
[23:0]	Reserved	Needs to be set to 24'd0

15.16 READ

"Read" instruction.

Used for reading the normal pages of embedded Flash.

15.16.1 7-Bit Addressing

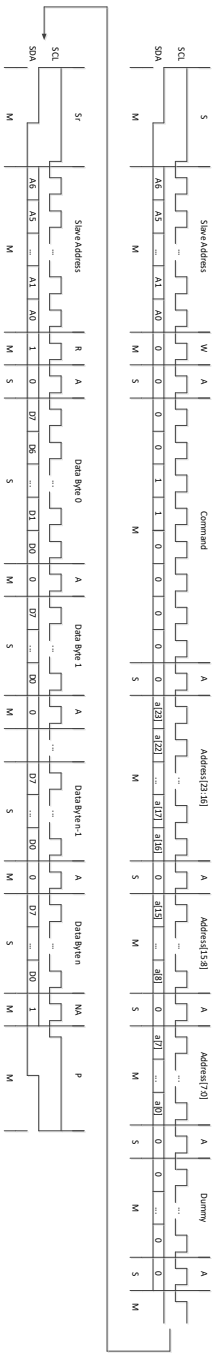

Figure 15-37 READ

15.16.1.1 1K/2K/4K/7K

Table 15-9 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

15.16.1.2 10K

Table 15-10 Address

Address	Item	Description
[23:20]	Reserved	Must be set to 4'b0000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[9:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

15.16.2 10-bit addressing

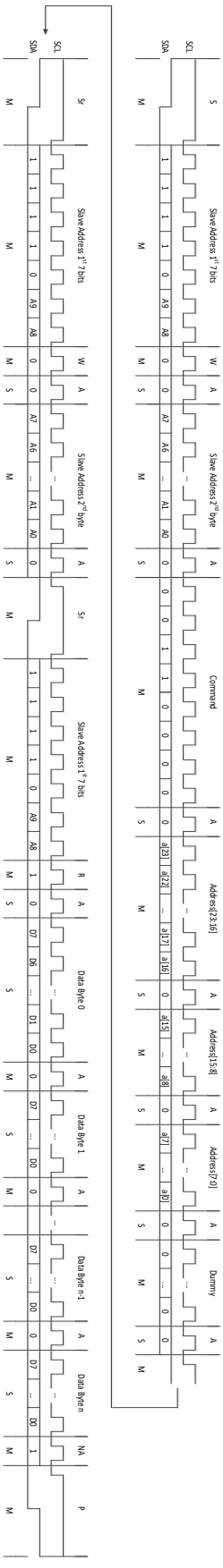

Figure 15-38 READ

15.16.2.1 1K/2K/4K/7K

Table 15-11 Address

Address	Item	Description
[23:19]	Reserved	Must be set to 5'b00000
[18:17]	Ba[1:0]	Heap Address
[16:8]	Ra[8:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

15.16.2.2 10K

Table 15-12 Address

Address	Item	Description
[23:20]	Reserved	Must be set to 4'b0000
[19:18]	Ba[1:0]	Heap Address
[17:8]	Ra[9:0]	Page Address
[7:6]	Reserved	Must be set to 2'b00
[5:0]	Ca[5:0]	32-bit data page internal offset address

15.17 READ_CTL

"Control Read" instruction.

Used for reading the feature control bits of embedded Flash.

15.17.1 7-Bit Addressing

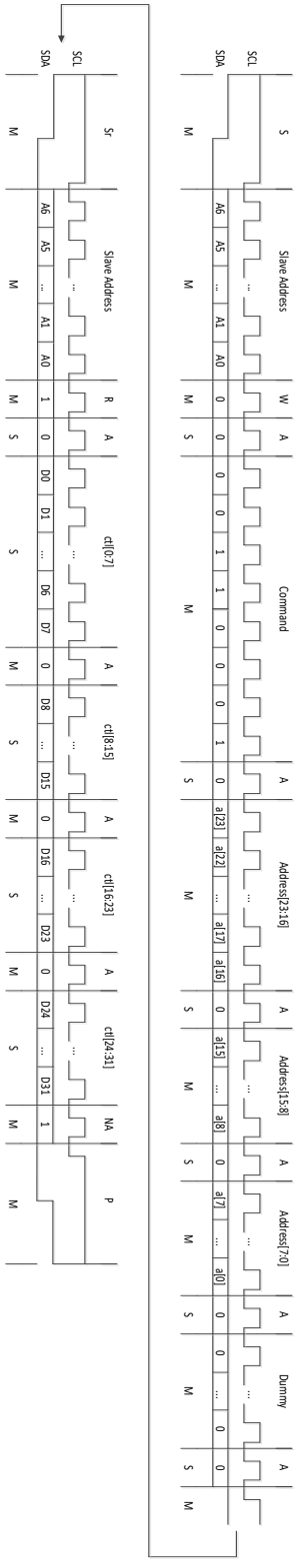

Figure 15-39 READ_CTL

Table 15-13 Address

Address	Item	Description
[23:0]	Reserved	Needs to be set to 24'd0

15.17.2 10-bit addressing

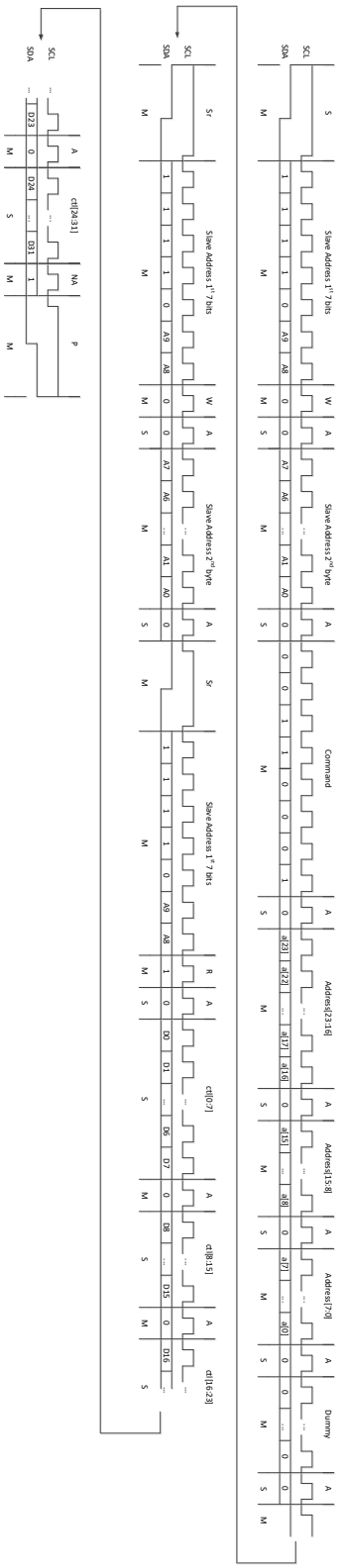

Figure 15-40 READ_CTL

Table 15-14 Address

Address	Item	Description

Address	Item	Description
[23:0]	Reserved	Needs to be set to 24'd0

15.18 PROGRAM_LOCK

"Lock" instruction.

Used to disable "Program" and "Read" operations of the bitstream.

Data sequentially shifts into the address of the last bitstream page "lock_ra[0:8]", heap address "lock_ba[0:1]", and lock flag "lock" and "4'd0".

For 1K devices, lock_ba[1:0] is the reserved bit.

For 2K devices, lock_ba[1:0] is the reserved heap address and should be set to 2'b11.

15.18.1 7-Bit Addressing

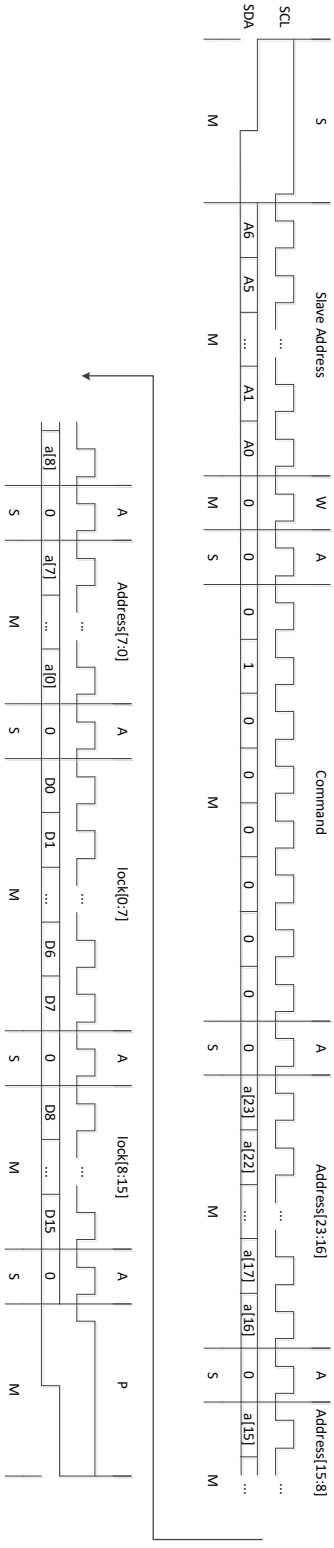

Figure 15-41 PROGRAM_LOCK

Table 15-15 Address

Address	**Item**	**Description**
[23:0]	Reserved	Needs to be set to 24'd0

15.18.2 10-bit addressing

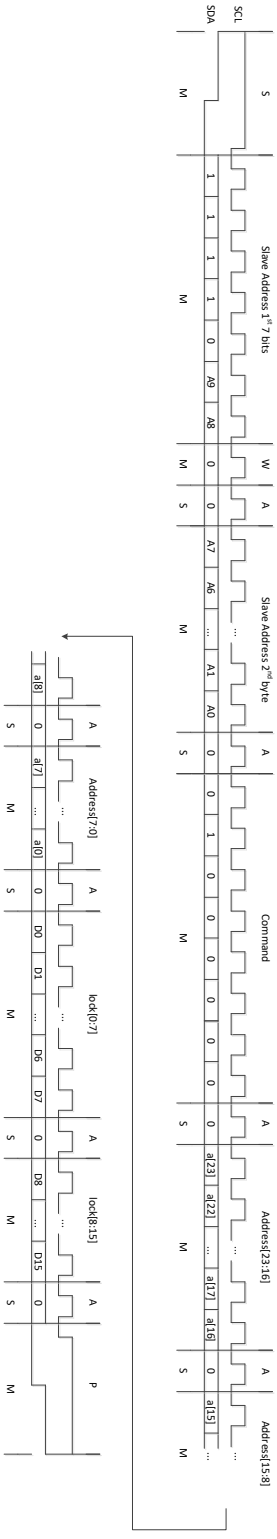

Figure 15-42 PROGRAM_LOCK

Table 15-16 Address

Address	Item	Description
[23:0]	Reserved	Needs to be set to 24'd0

15.19 EFlash_SLEEP

"Embedded Flash Sleep" instruction.

Used for putting embedded Flash into sleep mode.

15.19.1 7-Bit Addressing

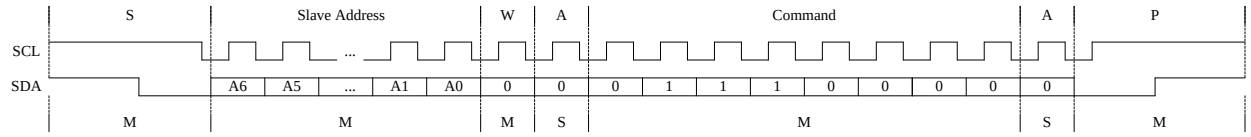

Figure 15-43 EFlash_SLEEP

15.19.2 10-bit addressing

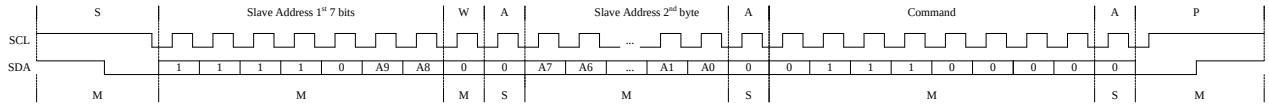

Figure 15-44 EFlash_SLEEP

15.20 EFlash_WAKEUP

"Embedded Flash Wake-up" instruction.

Used for waking up embedded Flash.

15.20.1 7-Bit Addressing

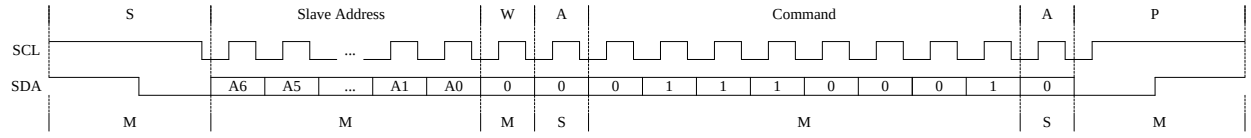

Figure 15-45 EFlash_WAKEUP

15.20.2 10-bit addressing

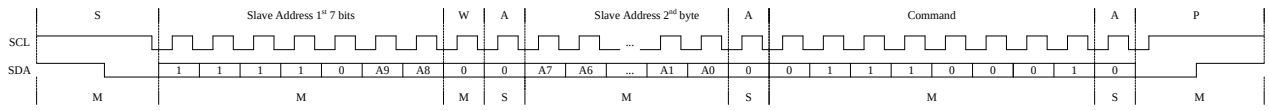

Figure 15-46 EFlash_WAKEUP

15.21 ISC_ENABLE

"ISC Enable" instruction.

15.21.1 7-Bit Addressing

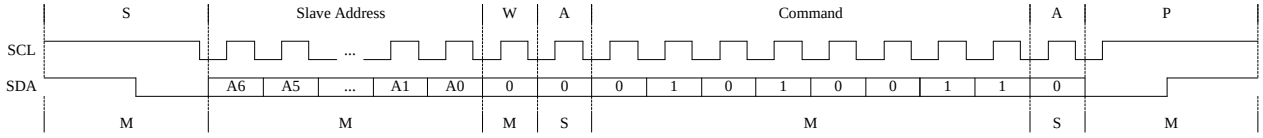

Figure 15-47 ISC_ENABLE

15.21.2 10-bit addressing

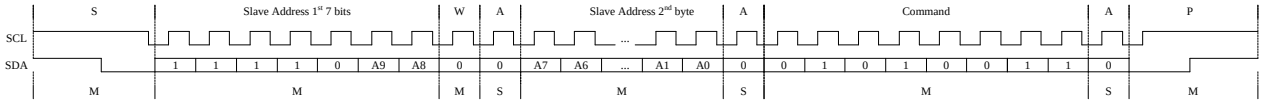

Figure 15-48 ISC_ENABLE

15.22 ISC_DISABLE

"ISC Disable" instruction.

15.22.1 7-Bit Addressing

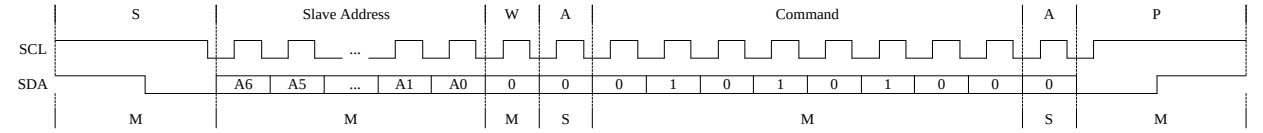

Figure 15-49 ISC_DISABLE

15.22.2 10-bit addressing

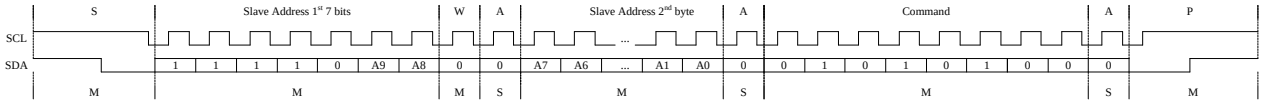

Figure 15-50 ISC_DISABLE

15.23 ISC_ENABLE_MSPI

Preserves the "ISC Enable" instruction for the master SPI interface.

15.23.1 7-Bit Addressing

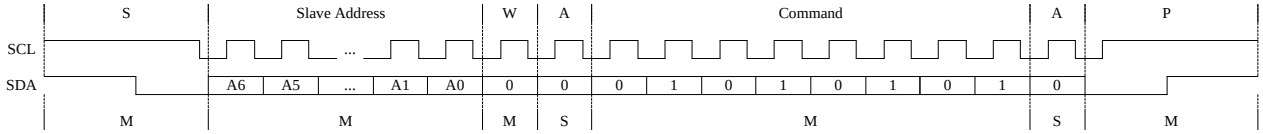

Figure 15-51 ISC_ENABLE_MSPI

15.23.2 10-bit addressing

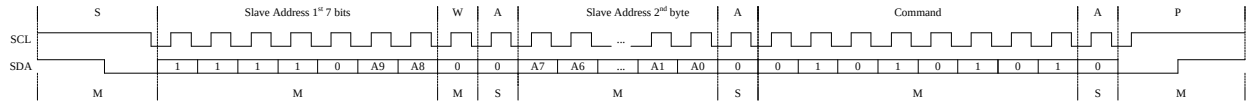

Figure 15-52 ISC_ENABLE_MSPI

15.24 ISC_DISABLE_MSPI

Reserve the "ISC Disable" instruction for the master SPI interface.

15.24.1 7-Bit Addressing

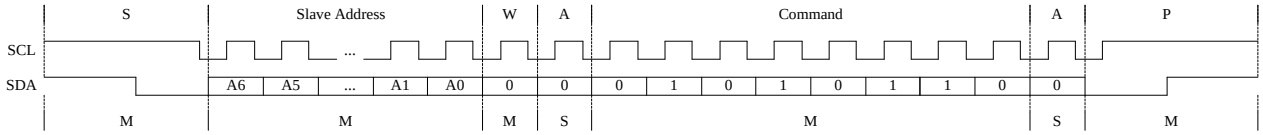

Figure 15-53 ISC_DISABLE_MSPI

15.24.2 10-bit addressing

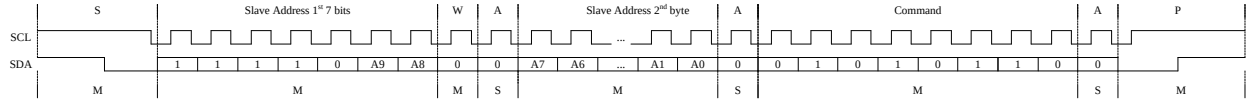

Figure 15-54 ISC_DISABLE_MSPI

Chapter 16 Appendix 6

Description of internal slave APB interface registers.

16.1 IDCODE Register 0

Table 16-1 IDCODE Register 0

Bit	Item	Description
[7:0]	idcode[7:0]	IDCODE[7:0]

16.2 IDCODE Register 1

Table 16-2 IDCODE Register 1

Bit	Item	Description
[7:0]	idcode[15:8]	IDCODE[15:8]

16.3 IDCODE Register 2

Table 16-3 IDCODE Register 2

Bit	Item	Description
[7:0]	idcode[23:16]	IDCODE[23:16]

16.4 IDCODE Register 3

Table 16-4 IDCODE Register 3

Bit	Item	Description
[7:0]	idcode[31:24]	IDCODE[31:24]

16.5 USERCODE Register 0

Table 16-5 USERCODE Register 0

Bit	Item	Description
[7:0]	usercode[7:0]	USERCODE[7:0]

16.6 USERCODE Register 1

Table 16-6 USERCODE Register 1

Bit	Item	Description
[7:0]	usercode[15:8]	USERCODE[15:8]

16.7 USERCODE Register 2

Table 16-7 USERCODE Register 2

Bit	Item	Description
[7:0]	usercode[23:16]	USERCODE[23:16]

16.8 USERCODE Register 3

Table 16-8 USERCODE Register 3

Bit	Item	Description
[7:0]	usercode[31:24]	USERCODE[31:24]

16.9 UID Register 0

Table 16-9 UID Register 0

Bit	Item	Description
[7:0]	uid[7:0]	UID[7:0]

16.10 UID Register 1

Table 16-10 UID Register 1

Bit	Item	Description
[7:0]	uid[15:8]	UID[15:8]

16.11 UID Register 2

Table 16-11 UID Register 2

Bit	Item	Description
[7:0]	uid[23:16]	UID[23:16]

16.12 UID Register 3

Table 16-12 UID Register 3

Bit	Item	Description
[7:0]	uid[31:24]	UID[31:24]

16.13 UID Register 4

Table 16-13 UID Register 4

Bit	Item	Description
[7:0]	uid[39:32]	UID[39:32]

16.14 UID Register 5

Table 16-14 UID Register 5

Bit	Item	Description
[7:0]	uid[47:40]	UID[47:40]

16.15 UID Register 6

Table 16-15 UID Register 6

Bit	Item	Description
[7:0]	uid[55:48]	UID[55:48]

16.16 UID Register 7

Table 16-16 UID Register 7

Bit	Item	Description
[7:0]	uid[63:56]	UID[63:56]

16.17 CCS Status Register 0

Table 16-17 CCS Status Register 0

Bit	Item	Description
[7:0]	status[7:0]	CCS Status Register [7:0]

16.18 CCS Status Register 1

Table 16-18 CCS Status Register 1

Bit	Item	Description
[7:0]	status[15:8]	CCS Status Register [15:8]

16.19 CCS Status Register 2

Table 16-19 CCS Status Register 2

Bit	Item	Description
[7:0]	status[23:16]	CCS Status Register [23:16]

16.20 CCS Status Register 3

Table 16-20 CCS Status Register3

Bit	Item	Description
[7:0]	status[31:24]	CCS Status Register [31:24]

16.21 Interrupt Control Register

Table 16-21 Interrupt Control Registers

Bit	Item	Description
[7:5]	Reserved	
[4]	irq_rrdy	Read Ready Interrupt Enable 0: Disabled 1: Enabled
[3]	irq_wrdy	Write Ready Interrupt Enable 0: Disabled 1: Enabled
[2]	irq_idle	embedded Flash Idle Interrupt Enable 0: Disabled 1: Enabled
[1]	irq_si2c_proc	Slave I^2C Interface Operation Interrupt Enable 0: Disabled 1: Enabled
[0]	irq_sspi_proc	Slave SPI Interface Operation Interrupt Enable 0: Disabled 1: Enabled

16.22 Command register

Table 16-22 Command Register

Instruction	Description	Op Code
NOP	No Operation	FF
WREN	Write Enable	51
WRDIS	Write Disable	52
RESET	Reset CPLD Devices	60
ERASE	Erase Bulk	10
ERASE_PAGE	Erase Page	11
ERASE_CTL	Erase Feature Control bit	12
PROGRAM	Program Page	20
PROGRAM_CTL	Program Feature Control Bit	21
READ	Read	30
READ_CTL	Read Feature Control Page	31
PROGRAM_LOCK	Lock Embedded Flash	40
EFlash_SLEEP	Embedded Flash Sleep	70
EFlash_WAKEUP	Embedded Flash Wake Up	71

16.23 Address Register 0

Table 16-23 Address Register 0

Bit	Item	Description
[7:6]	Reserved	
[5:0]	Ca[5:0]	32-bit data page internal offset address

16.24 Address Register 1

Table 16-24 Address Register 1

Bit	Item	Description
[7:0]	Ra[7:0]	Page Address [7:0]

16.25 Address Register 2

16.25.1 1K/2K/4K/7K

Table 16-25 Address Register 2

Bit	Item	Description
[7:3]	Reserved	
[2:1]	Ba[1:0]	Heap Address

Bit	Item	Description
[0]	Ra[8]	Page Address [8]

For 1K devices, Ba[1:0] are reserved bits.

For 2K devices, Ba[1:0] are reserved heap addresses and should be set to 2'b11.

16.25.2 10K

Table 16-26 Address Register 2

Bit	Item	Description
[7:4]	Reserved	
[3:2]	Ba[1:0]	Heap Address
[1:0]	Ra[9:8]	Page Address [9:8]

16.26 Data Transmit Registers

Used for sending embedded Flash programming data.

16.27 Data Receive Registers

Used for receiving embedded Flash read data.

16.28 Status register

Table 16-27 Status Register

Bit	Item	Description
[7:5]	Reserved	
[4]	rrdy	Read status indication 0: Not ready to read data 1: Ready to read data and can read data
[3]	wrdy	Program status indication 0: Not ready to receive programming data 1: Ready to receive programming data and can write programming data
[2]	busy	embedded Flash busy indication 0: Idle 1: Busy
[1]	si2c_proc	Slave I^2C interface status 0: Idle 1: Operating
[0]	sspi_proc	Slave SPI interface status 0: Idle 1: Operating

16.29 Interrupt Status Registers

Table 16-28 Interrupt Status Register

Bit	Item	Description
[7:5]	Reserved	
[4]	irq_rrdy	Read ready interrupt, written to 1 for clear 0: Not ready to read data 1: Ready to read data and can read data
[3]	irq_wrdy	Write ready interrupt, written to 1 for clear 0: Not ready to receive programming data 1: Ready to receive programming data and can write programming data
[2]	irq_idle	embedded Flash idle interrupt, written to 1 for clear
[1]	irq_si2c_proc	Slave I^2C interface operation interrupt, written to 1 for clear
[0]	irq_sspi_proc	Slave SPI interface operation interrupt, written to 1 for clear

16.30 Interrupt Registers

Table 16-29 Interrupt Register

Bit	Item	Description
[7:5]	Reserved	
[4]	irq_timer	Timer functional module interrupt
[3]	irq_si2c1	I^2C functional module 1 interrupt
[2]	irq_si2c0	I^2C functional module 2 interrupt
[1]	irq_spi	SPI functional module interrupt
[0]	irq_ccs	embedded Flash controller interrupt

16.31 Lock Page Address 0 Register

Table 16-30 Lock Page Address 0 Register

Bit	Item	Description
[7:0]	lock_ra[7:0]	Bitstream tail page address lower 8bits

16.32 Lock Page Address 1 Register

16.32.1 1K/2K/4K/7K

Table 16-31 Lock Page Address 1 Register

Bit	Item	Description
[7:4]	Reserved	
[3]	lock	Lock flag bits
[2:1]	lock_ba[1:0]	Bitstream tail page heap address
[0]	lock_ra[8]	Bitstream tail page heap address higher 1bit

16.32.2 10K

Table 16-32 Lock Page Address 1 Register

Bit	Item	Description
[7:5]	Reserved	
[4]	lock	Lock flag bits
[3:2]	lock_ba[1:0]	Bitstream tail page heap address
[1:0]	lock_ra[9:8]	Bitstream tail page address higher 2bits

Chapter 17 Appendix 7

Internal Slave APB Operation Process.

17.1 Read IDCODE

Table 17-1 Read IDCODE

Steps	Operation Description
1	Read IDCODE Register 0
2	Read IDCODE Register 1
3	Read IDCODE Register 2
4	Read IDCODE Register 3

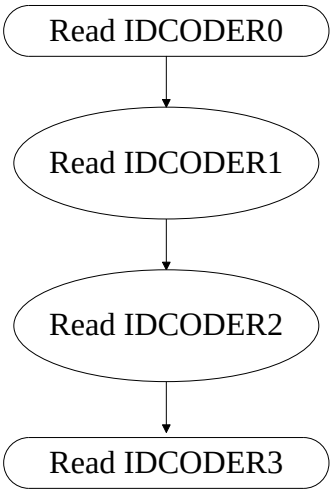

Figure 17-1 Read IDCODE

17.2 Read USERCODE

Table 17-2 Read USERCODE

Steps	Operation Description
1	Read USERCODE Register 0
2	Read USERCODE Register 1
3	Read USERCODE Register 2
4	Read USERCODE Register 3

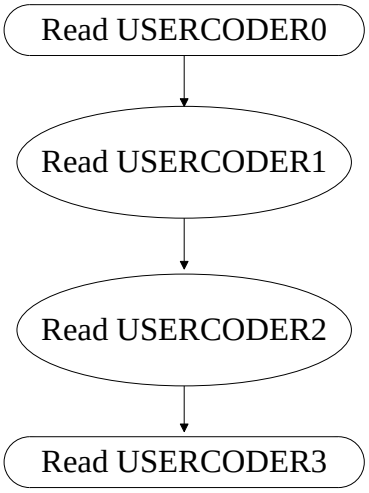

Figure 17-2 Read USERCODE

17.3 Read UID

Table 17-3 Read UID

Steps	Operation Description
1	Read UID Register 0
2	Read UID Register 1
3	Read UID Register 2
4	Read UID Register 3
5	Read UID Register 4
6	Read UID Register 5
7	Read UID Register 6
8	Read UID Register 7

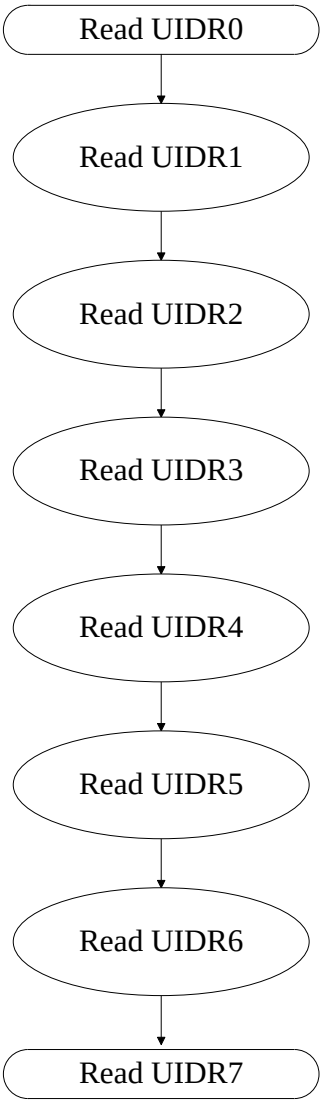

Figure 17-3 Read UID

17.4 Read CCS Status Register

Table 17-4 Read CCS Status Registers

Steps	Operation Description
1	Read CCS Status Register 0
2	Read CCS Status Register 1
3	Read CCS Status Register 2
4	Read CCS Status Register 3

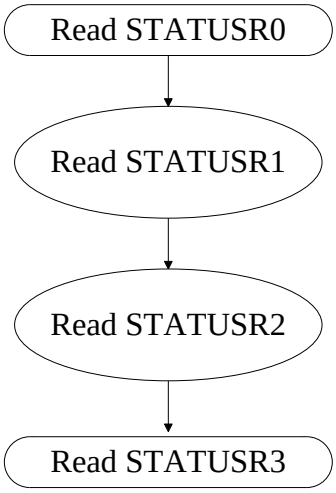

Figure 17-4 Read CCS Status Registers

17.5 Reset

Table 17-5 Reset

Steps	Operation Description
1	Write the "RESET" command to the command register

Figure 17-5 Reset

17.6 Put embedded Flash into Sleep Mode

This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.6.1 Peer to Peer Mode

Table 17-6 Put embedded Flash into Sleep Mode

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "EFlash_SLEEP" command to the command register
4	Continuously read the status register until 'busy' is detected low
5	Write the "WRDIS" command to the command register
6	Write the "NOP" command to the command register

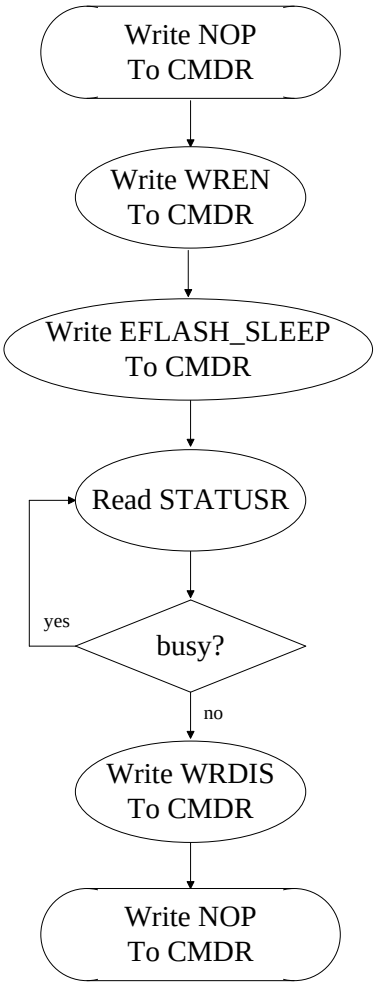

Figure 17-6 Put embedded Flash into Sleep Mode

17.6.2 Interrupt Mode

Table 17-7 Put embedded Flash into Sleep Mode

Steps	**Operation Description**
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "EFlash_SLEEP" command to the command register
4	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
5	Write to the interrupt status register to clear the embedded Flash idle interrupt
6	Write the "WRDIS" command to the command register
7	Write the "NOP" command to the command register

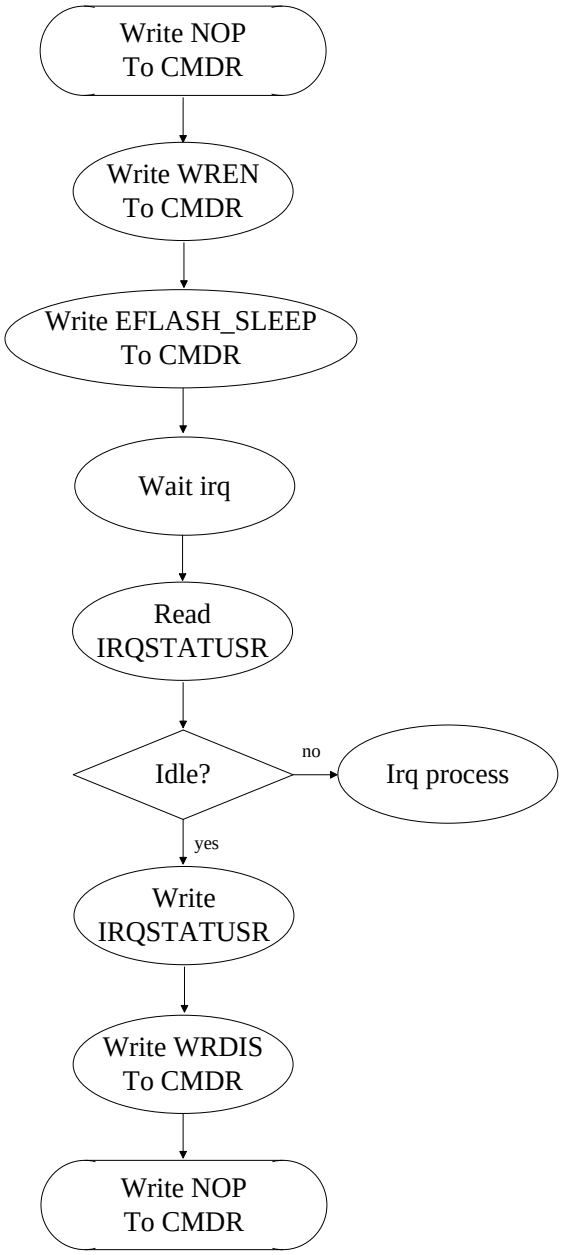

Figure 17-7 Put embedded Flash into Sleep Mode

17.7 Wake Up embedded Flash

This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.7.1 Peer to Peer Mode

Table 17-8 Wake Up embedded Flash

Steps	**Operation Description**
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register

Steps	Operation Description
3	Write the "EFlash_WAKEUP" command to the command register
4	Continuously read the status register until 'busy' is detected low
5	Write the "WRDIS" command to the command register
6	Write the "NOP" command to the command register

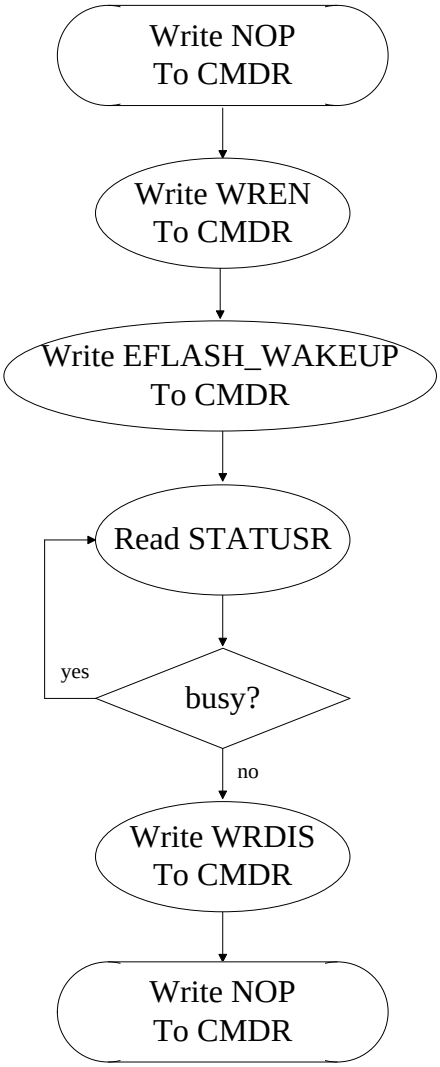

Figure 17-8 Wake Up embedded Flash

17.7.2 Interrupt Mode

Table 17-9 Wake Up embedded Flash

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "EFlash_WAKEUP" command to the command register
4	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt

Steps	Operation Description
5	Write to the interrupt status register to clear the embedded Flash idle interrupt
6	Write the "WRDIS" command to the command register

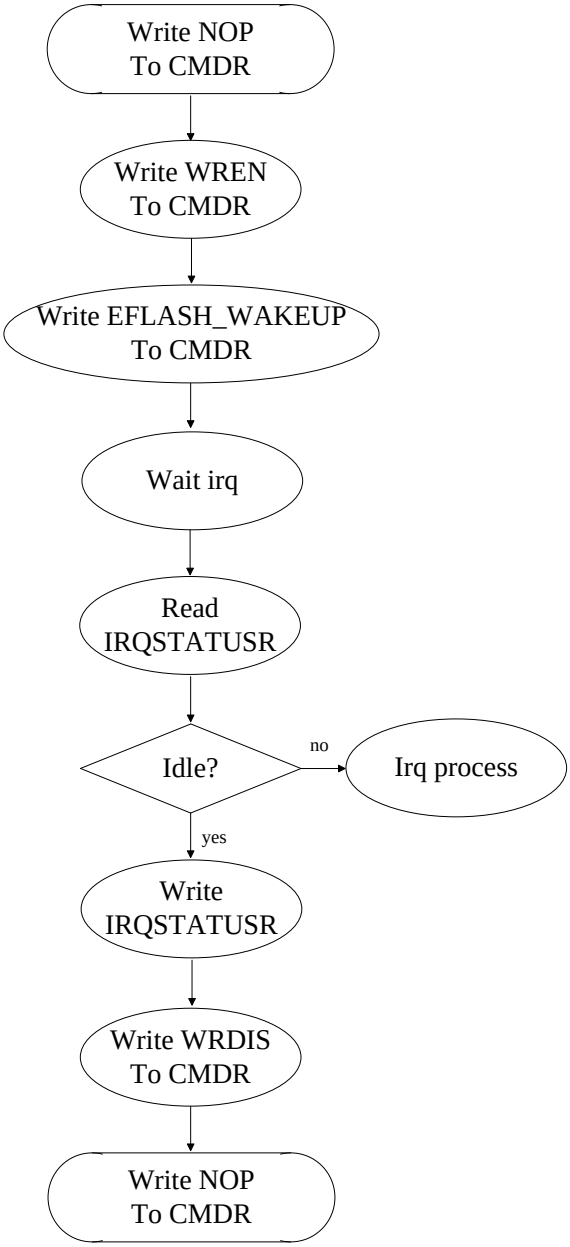

Figure 17-9 Wake Up embedded Flash

17.8 Program Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.8.1 Peer to Peer Mode

Table 17-10 Program Feature Control Bits

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "PROGRAM_CTL" command to the command register
4	Write feature control bits ctl[7:0] to the data transmit register
5	Write feature control bits ctl[15:8] to the data transmit register
6	Write feature control bits ctl[23:16] to the data transmit register
7	Continuously read the status register until 'busy' is detected low
8	Write the "READ_CTL" command to the command register
9	Read feature control bits ctl[7:0] from the data receive register
10	Read feature control bits ctl[15:8] from the data receive register
11	Read feature control bits ctl[23:16] from the data receive register
12	Write the "NOP" command to the command register
13	Continuously read the status register until 'busy' is detected low
14	Write the "WRDIS" command to the command register
15	Write the "NOP" command to the command register

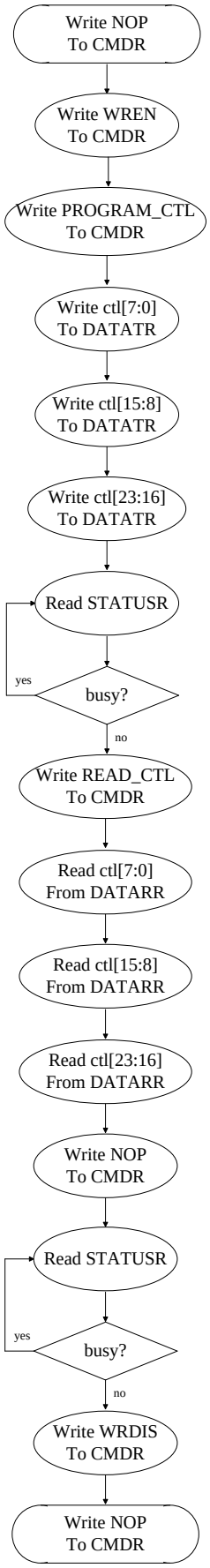

Figure 17-10 Program Feature Control Bits

17.8.2 Interrupt Mode

Table 17-11 Program Feature Control Bits

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "PROGRAM_CTL" command to the command register
4	After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt
5	Write feature control bits ctl[7:0] to the data transmit register
6	Write feature control bits ctl[15:8] to the data transmit register
7	Write feature control bits ctl[23:16] to the data transmit register
8	Write to the interrupt status register to clear the embedded Flash write ready interrupt
9	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
10	Write to the interrupt status register to clear the embedded Flash idle interrupt
11	Write the "READ_CTL" command to the command register
12	After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt
13	Read feature control bits ctl[7:0] from the data receive register
14	Read feature control bits ctl[15:8] from the data receive register
15	Read feature control bits ctl[23:16] from the data receive register
16	Write the "NOP" command to the command register
17	Write to the interrupt status register to clear the embedded Flash read ready interrupt
18	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
19	Write to the interrupt status register to clear the embedded Flash idle interrupt
20	Write the "WRDIS" command to the command register
21	Write the "NOP" command to the command register

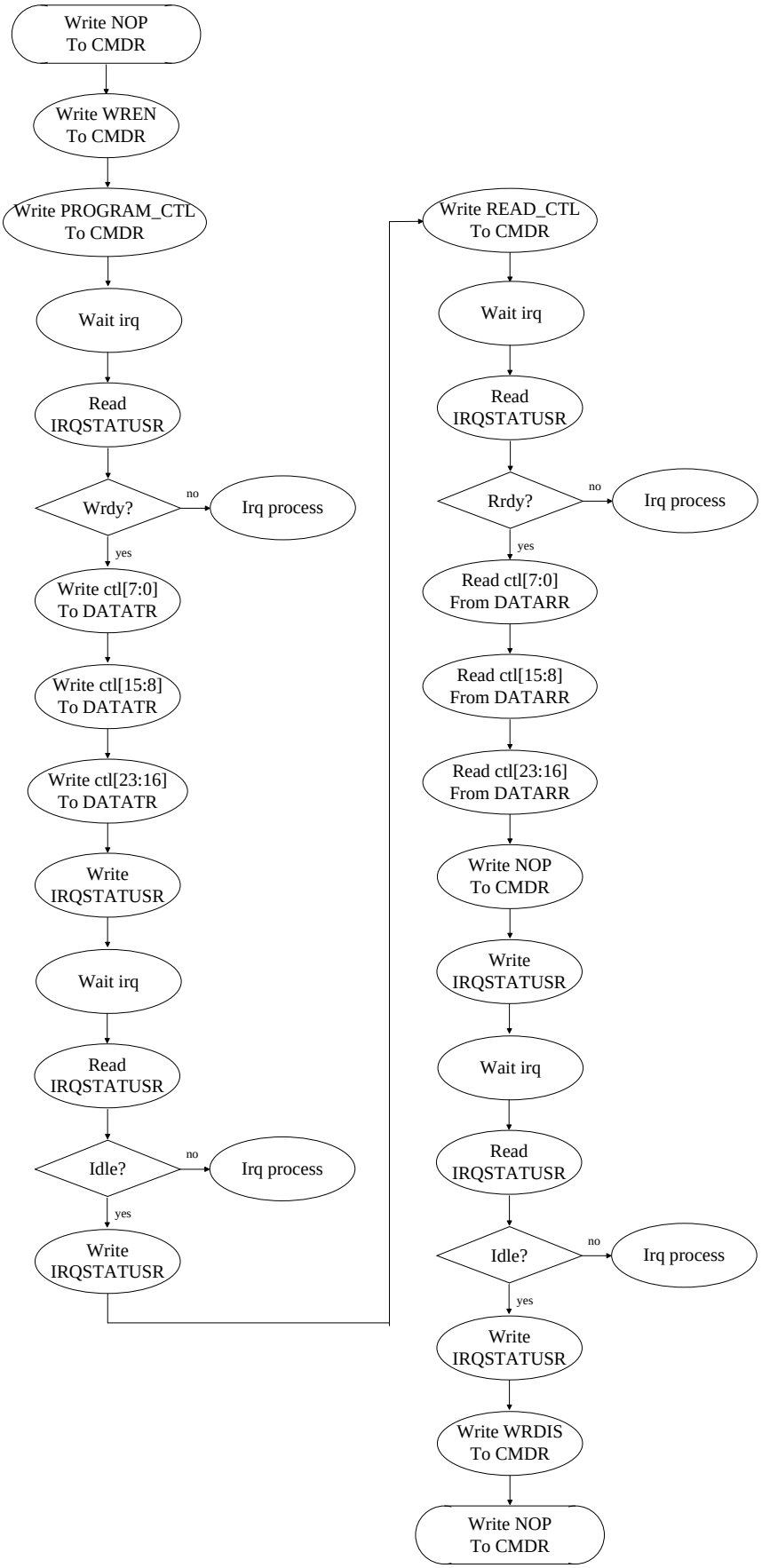

Figure 17-11 Program Feature Control Bits

17.9 Reset Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.9.1 Peer to Peer Mode

Table 17-12 Reset Feature Control Bits

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "ERASE_CTL" command to the command register
4	Continuously read the status register until 'busy' is detected low
5	Write the "READ_CTL" command to the command register
6	Read feature control bits ctl[7:0] from the data receive register
7	Read feature control bits ctl[15:8] from the data receive register
8	Read feature control bits ctl[23:16] from the data receive register
9	Write the "NOP" command to the command register
10	Continuously read the status register until 'busy' is detected low
11	Write the "WRDIS" command to the command register
12	Write the "NOP" command to the command register

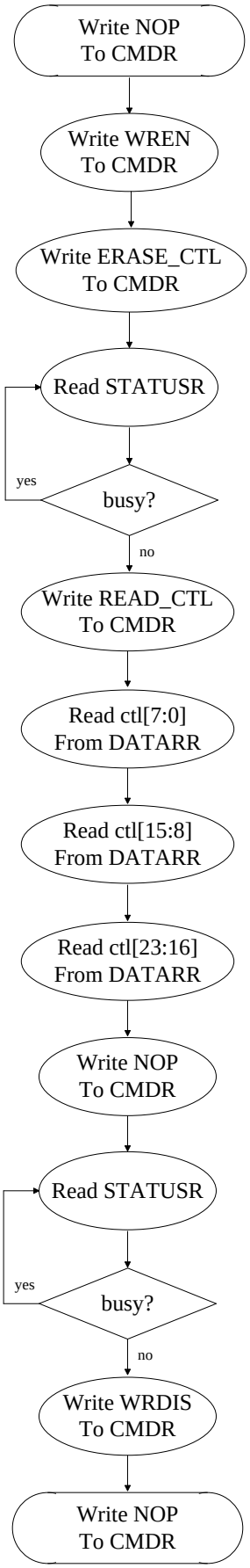

Figure 17-12 Reset Feature Control Bits

17.9.2 Interrupt Mode

Table 17-13 Reset Feature Control Bits

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "ERASE_CTL" command to the command register
4	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
5	Write to the interrupt status register to clear the embedded Flash idle interrupt
6	Write the "READ_CTL" command to the command register
7	After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt
8	Read feature control bits ctl[7:0] from the data receive register
9	Read feature control bits ctl[15:8] from the data receive register
10	Read feature control bits ctl[23:16] from the data receive register
11	Write the "NOP" command to the command register
12	Write to the interrupt status register to clear the embedded Flash read ready interrupt
13	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
14	Write to the interrupt status register to clear the embedded Flash idle interrupt
15	Write the "WRDIS" command to the command register
16	Write the "NOP" command to the command register

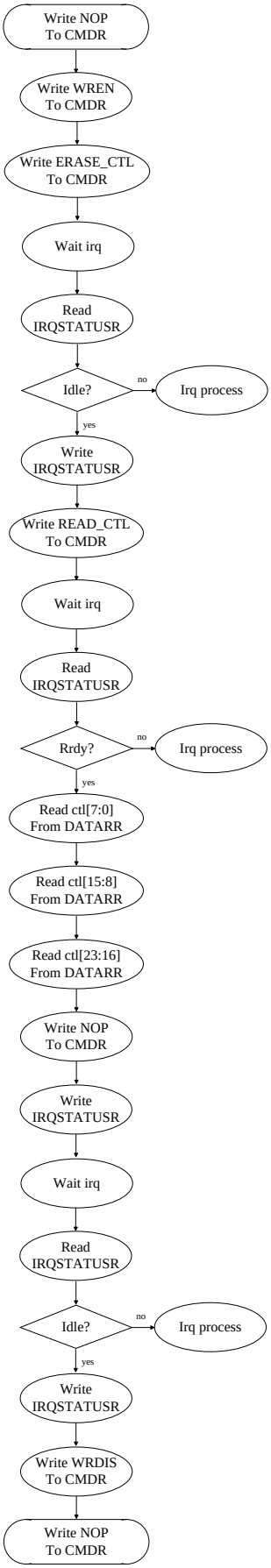

Figure 17-13 Reset Feature Control Bits

17.10 Read Feature Control Bits

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.10.1 Peer to Peer Mode

Table 17-14 Read Feature Control Bits

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "READ_CTL" command to the command register
4	Read feature control bits ctl[7:0] from the data receive register
5	Read feature control bits ctl[15:8] from the data receive register
6	Read feature control bits ctl[23:16] from the data receive register
7	Write the "NOP" command to the command register
8	Continuously read the status register until 'busy' is detected low
9	Write the "WRDIS" command to the command register
10	Write the "NOP" command to the command register

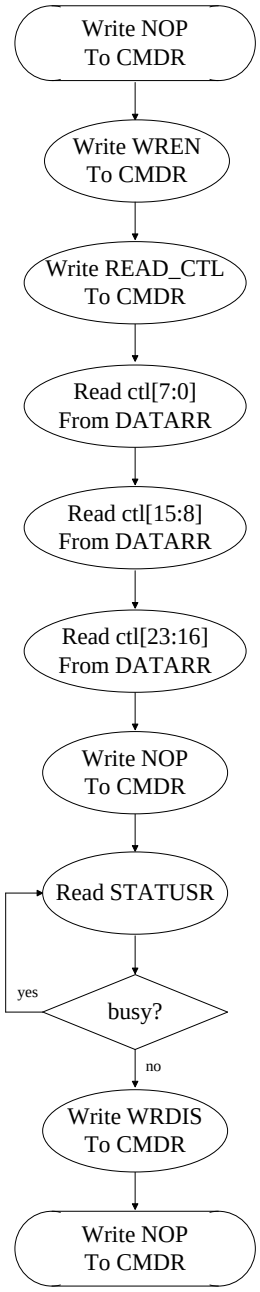

Figure 17-14 Read Feature Control Bits

17.10.2 Interrupt Mode

Table 17-15 Read Feature Control Bits

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "READ_CTL" command to the command register
4	After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt
5	Read feature control bits ctl[7:0] from the data receive register
6	Read feature control bits ctl[15:8] from the data receive register
7	Read feature control bits ctl[23:16] from the data receive register
8	Write the "NOP" command to the command register
9	Write to the interrupt status register to clear the embedded Flash read ready interrupt
10	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
11	Write to the interrupt status register to clear the embedded Flash idle interrupt
12	Write the "WRDIS" command to the command register
13	Write the "NOP" command to the command register

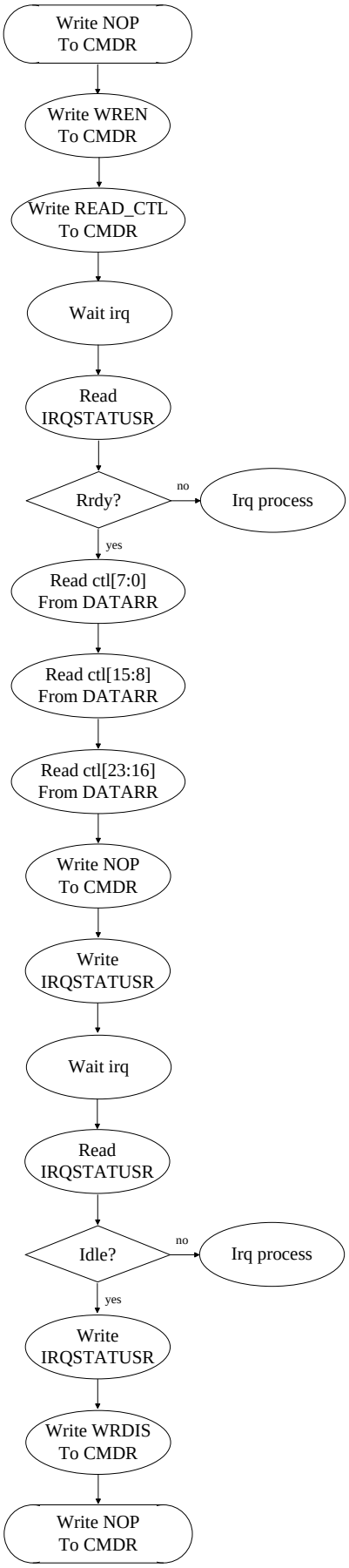

Figure 17-15 Read Feature Control Bits

17.11 Program Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.11.1 Peer to Peer Mode

Table 17-16 Program Bitstream

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the bitstream data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "PROGRAM" command to the command register
7	Write the first page of the bitstream data to the data transmit register
8	Continuously read the status register until 'busy' is detected low
9	Write the "READ" command to the command register
10	Read the first page of the bitstream data from the receive register
11	Write the "NOP" command to the command register
12	Continuously read the status register until 'busy' is detected low
13	Write to address register 0
14	Write to address register 1
15	Write the second page of the bitstream data to the address register 2
16	Write the "PROGRAM" command to the command register
17	Write the second page of the bitstream data to the data transmit register
18	Continuously read the status register until 'busy' is detected low
19	Write the "READ" command to the command register
20	Read the second page of the bitstream data from the data receive register
21	Write the "NOP" command to the command register
22	Continuously read the status register until 'busy' is detected low
23	……
24	Write to address register 0
25	Write to address register 1
26	Write the address of the nth page of the bitstream data to the address register 2
27	Write the "PROGRAM" command to the command register
28	Write the nth page of the bitstream data to the data transmit register
29	Continuously read the status register until 'busy' is detected low
30	Write the "READ" command to the command register
31	Read the nth page of the bitstream data from the data receive register
32	Write the "NOP" command to the command register

Steps	Operation Description
33	Continuously read the status register until 'busy' is detected low
34	Write the "WRDIS" command to the command register
35	Write the "NOP" command to the command register

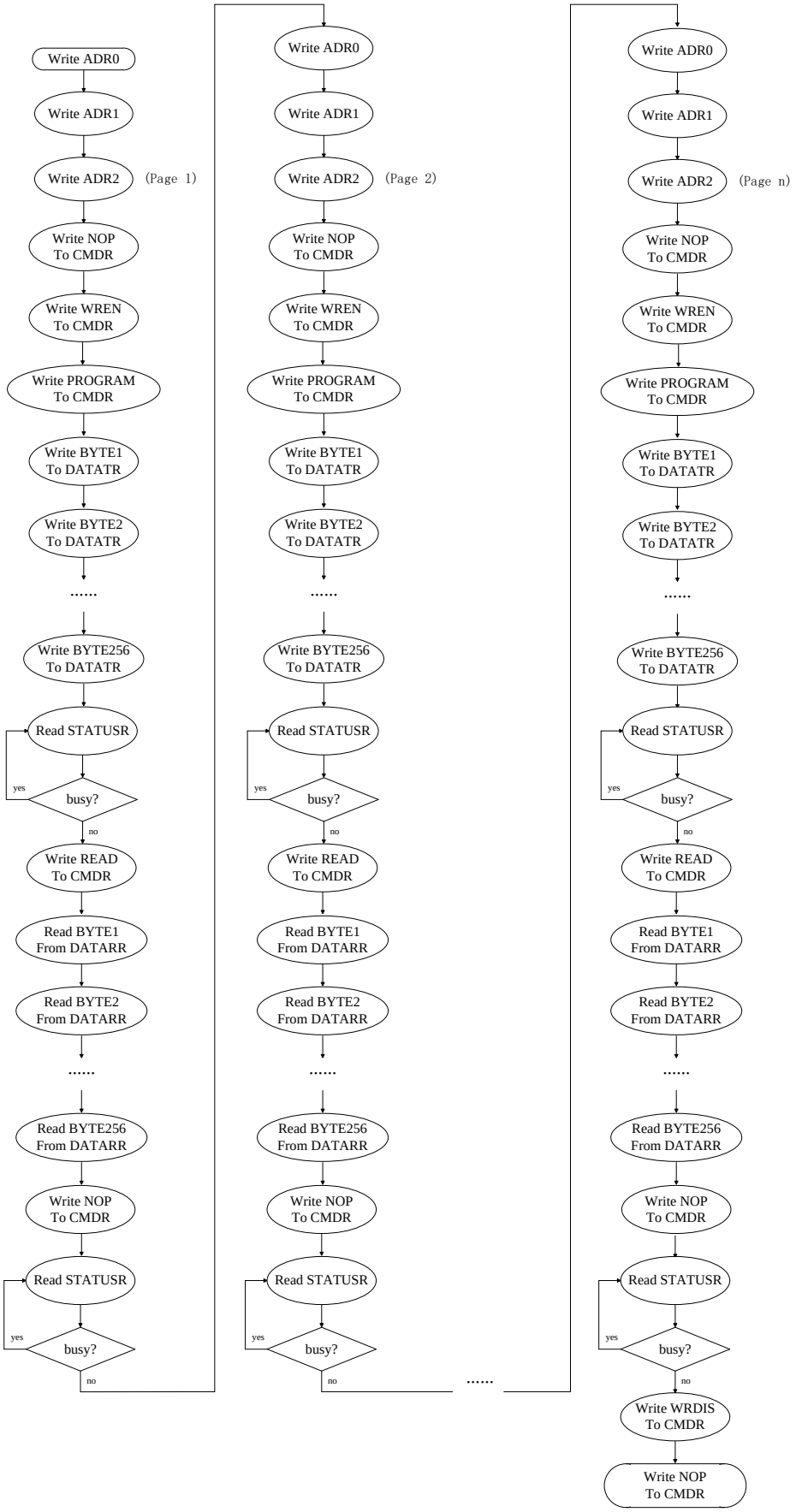

Figure 17-16 Program Bitstream

17.11.2 Interrupt Mode

Table 17-17 Program Bitstream

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the bitstream data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "PROGRAM" command to the command register
7	Write the first page of the bitstream data to the data transmit register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt Write the 1st byte to the data transmit register Write the 2nd byte to the data transmit register Write the 3rd byte to the data transmit register Write the 4th byte to the data transmit register Write to the interrupt status register to clear the embedded Flash write ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt Write the 5th byte to the data transmit register Write the 6th byte to the data transmit register Write the 7th byte to the data transmit register Write the 8th byte to the data transmit register Write to the interrupt status register to clear the embedded Flash write ready interrupt ..... After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt Write the 253rd byte to the data transmit register Write the 254th byte to the data transmit register Write the 255th byte to the data transmit register Write the 256th byte to the data transmit register Write to the interrupt status register to clear the embedded Flash write ready interrupt
8	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
9	Write to the interrupt status register to clear the embedded Flash idle interrupt
10	Write the "READ" command to the command register
11	Read the first page of the bitstream data from the data receive register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt

Steps	Operation Description
	Read the 253rd byte from the data receive register Read the 254th byte from the data receive register Read the 255th byte from the data receive register Read the 256th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt
12	Write the "NOP" command to the command register
13	Write to the interrupt status register to clear the embedded Flash read ready interrupt
14	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
15	Write to the interrupt status register to clear the embedded Flash idle interrupt
16	Write to address register 0
17	Write to address register 1
18	Write the second page of the bitstream data to the address register 2
19	Write the "PROGRAM" command to the command register
20	Write the second page of the bitstream data to the data transmit register
21	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
22	Write to the interrupt status register to clear the embedded Flash idle interrupt
23	Write the "READ" command to the command register
24	Read the second page of the bitstream data from the data receive register
25	Write the "NOP" command to the command register
26	Write to the interrupt status register to clear the embedded Flash read ready interrupt
27	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
28	Write to the interrupt status register to clear the embedded Flash idle interrupt
29	……
30	Write to address register 0
31	Write to address register 1
32	Write the address of the nth page of the bitstream data to the address register 2
33	Write the "PROGRAM" command to the command register
34	Write the nth page of the bitstream data to the data transmit register
35	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
36	Write to the interrupt status register to clear the embedded Flash idle interrupt
37	Write the "READ" command to the command register
38	Read the nth page of the bitstream data from the data receive register
39	Write the "NOP" command to the command register
40	Write to the interrupt status register to clear the embedded Flash read ready interrupt
41	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
42	Write to the interrupt status register to clear the embedded Flash idle interrupt
43	Write the "WRDIS" command to the command register
44	Write the "NOP" command to the command register

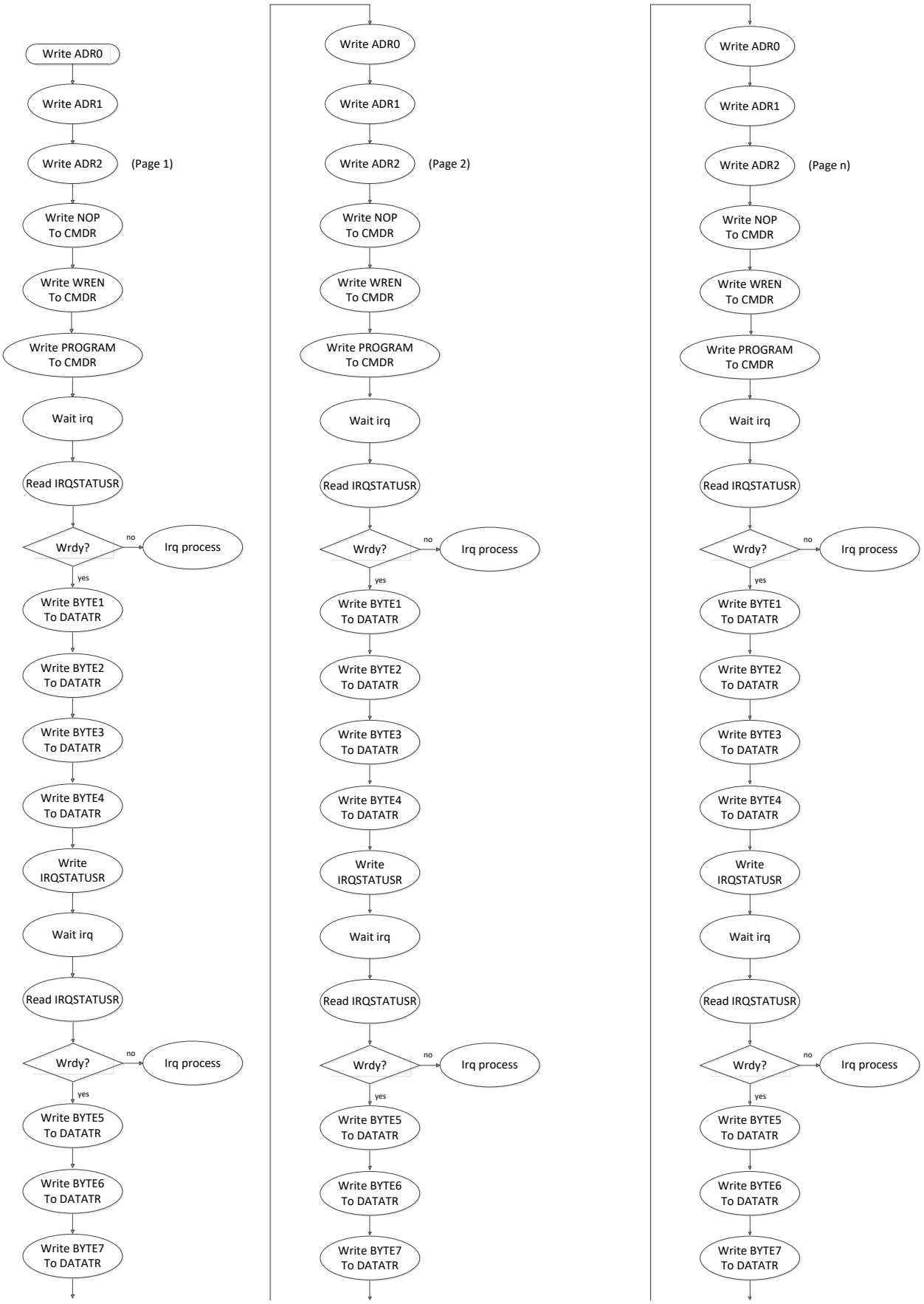

Figure 17-17 Program Bitstream 1

Figure 17-18 Program Bitstream 2

Figure 17-19 Program Bitstream 3

17.12 Erase Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.12.1 Peer to Peer Mode

Table 17-18 Erase Bitstream

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the bitstream data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "ERASE_PAGE" command to the command register
7	Continuously read the status register until 'busy' is detected low
8	Write the "READ" command to the command register
9	Read the first page of the bitstream data from the receive register
10	Write the "NOP" command to the command register
11	Continuously read the status register until 'busy' is detected low
12	Write to address register 0
13	Write to address register 1
14	Write the second page of the bitstream data to the address register 2
15	Write the "NOP" command to the command register
16	Write the "ERASE_PAGE" command to the command register
17	Continuously read the status register until 'busy' is detected low
18	Write the "READ" command to the command register
19	Read the second page of the bitstream data from the data receive register
20	Write the "NOP" command to the command register
21	Continuously read the status register until 'busy' is detected low
22	……
23	Write to address register 0
24	Write to address register 1
25	Write the address of the nth page of the bitstream data to the address register 2
26	Write the "NOP" command to the command register
27	Write the "ERASE_PAGE" command to the command register
28	Continuously read the status register until 'busy' is detected low
29	Write the "READ" command to the command register
30	Read the nth page of the bitstream data from the data receive register
31	Write the "NOP" command to the command register
32	Continuously read the status register until 'busy' is detected low

Steps	Operation Description
33	Write the "WRDIS" command to the command register
34	Write the "NOP" command to the command register

Figure 17-20 Erase Bitstream

17.12.2 Interrupt Mode

Table 17-19 Erase Bitstream

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the bitstream data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "ERASE_PAGE" command to the command register
7	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
8	Write to the interrupt status register to clear the embedded Flash idle interrupt
9	Write the "READ" command to the command register
10	Read the first page of the bitstream data from the data receive register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 253rd byte from the data receive register Read the 254th byte from the data receive register Read the 255th byte from the data receive register Read the 256th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt
11	Write the "NOP" command to the command register
12	Write to the interrupt status register to clear the embedded Flash read ready interrupt
13	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
14	Write to the interrupt status register to clear the embedded Flash idle interrupt
15	Write to address register 0
16	Write to address register 1
17	Write the second page of the bitstream data to the address register 2
18	Write the "ERASE_PAGE" command to the command register
19	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
20	Write to the interrupt status register to clear the embedded Flash idle interrupt
21	Write the "READ" command to the command register
22	Read the second page of the bitstream data from the data receive register

Steps	Operation Description
23	Write the "NOP" command to the command register
24	Write to the interrupt status register to clear the embedded Flash read ready interrupt
25	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
26	Write to the interrupt status register to clear the embedded Flash idle interrupt
27	……
28	Write to address register 0
29	Write to address register 1
30	Write the address of the nth page of the bitstream data to the address register 2
31	Write the "ERASE_PAGE" command to the command register
32	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
33	Write to the interrupt status register to clear the embedded Flash idle interrupt
34	Write the "READ" command to the command register
35	Read the nth page of the bitstream data from the data receive register
36	Write the "NOP" command to the command register
37	Write to the interrupt status register to clear the embedded Flash read ready interrupt
38	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
39	Write to the interrupt status register to clear the embedded Flash idle interrupt
40	Write the "WRDIS" command to the command register
41	Write the "NOP" command to the command register

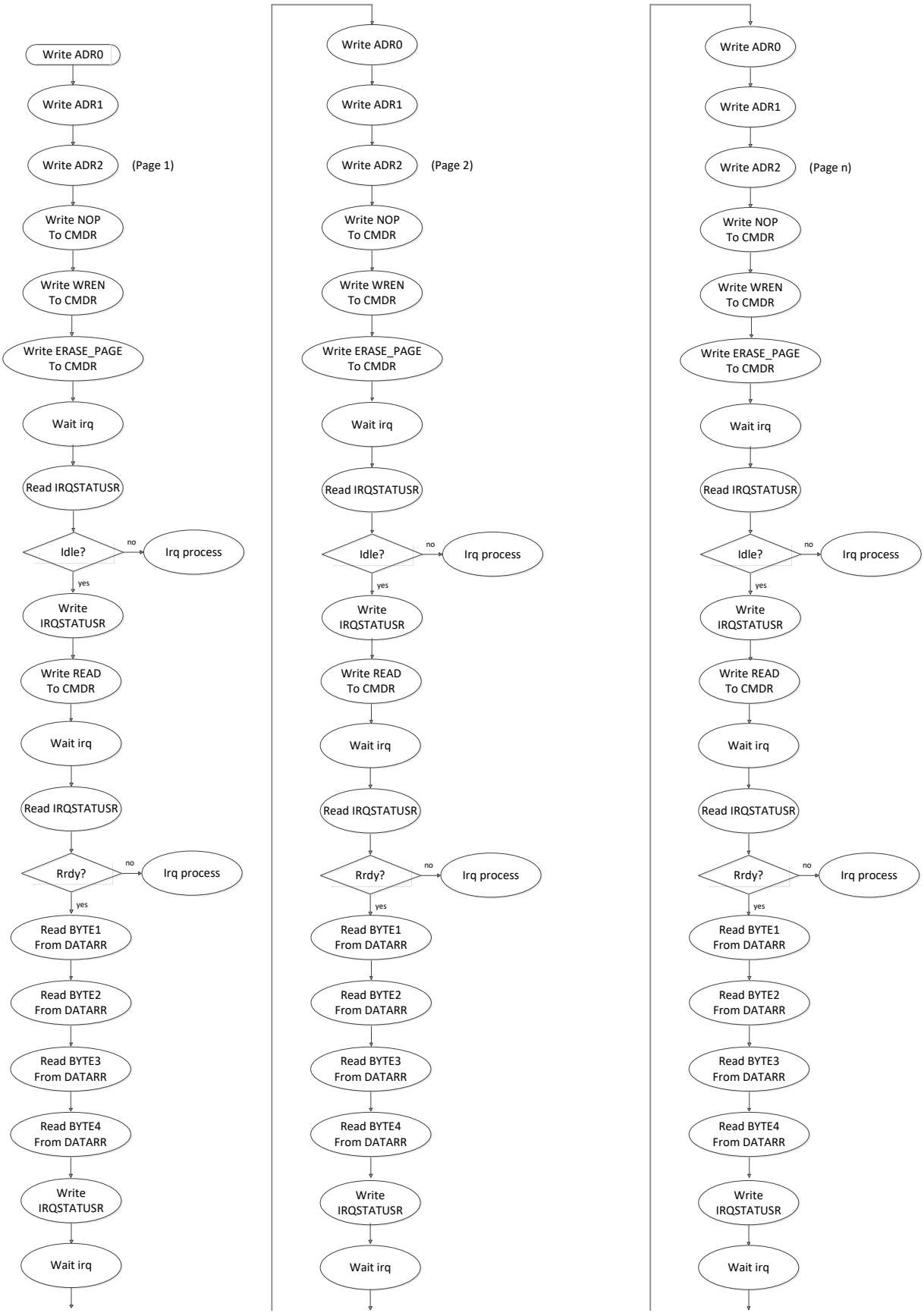

Figure 17-21 Erase Bitstream 1

Figure 17-22 Erase Bitstream 2

17.13 Read Bitstream

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.13.1 Peer to Peer Mode

Table 17-20 Read Bitstream

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the starting address of the bitstream in the embedded Flash to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "READ" command to the command register
7	Read the bitstream from the data receiver register
8	Write the "NOP" command to the command register
9	Continuously read the status register until 'busy' is detected low
10	Write the "WRDIS" command to the command register
11	Write the "NOP" command to the command register

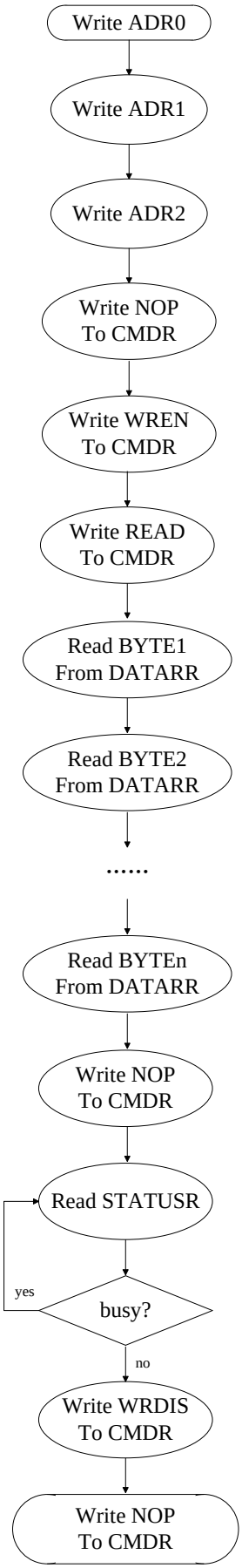

Figure 17-23 Read Bitstream

17.13.2 Interrupt Mode

Table 17-21 Read Bitstream

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the starting address of the bitstream in the embedded Flash to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "READ" command to the command register
7	Read the bitstream from the data receiver register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... Read the nth byte from the data receiver register
8	Write the "NOP" command to the command register
9	Write to the interrupt status register to clear the embedded Flash read ready interrupt
10	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
11	Write to the interrupt status register to clear the embedded Flash idle interrupt
12	Write the "WRDIS" command to the command register
13	Write the "NOP" command to the command register

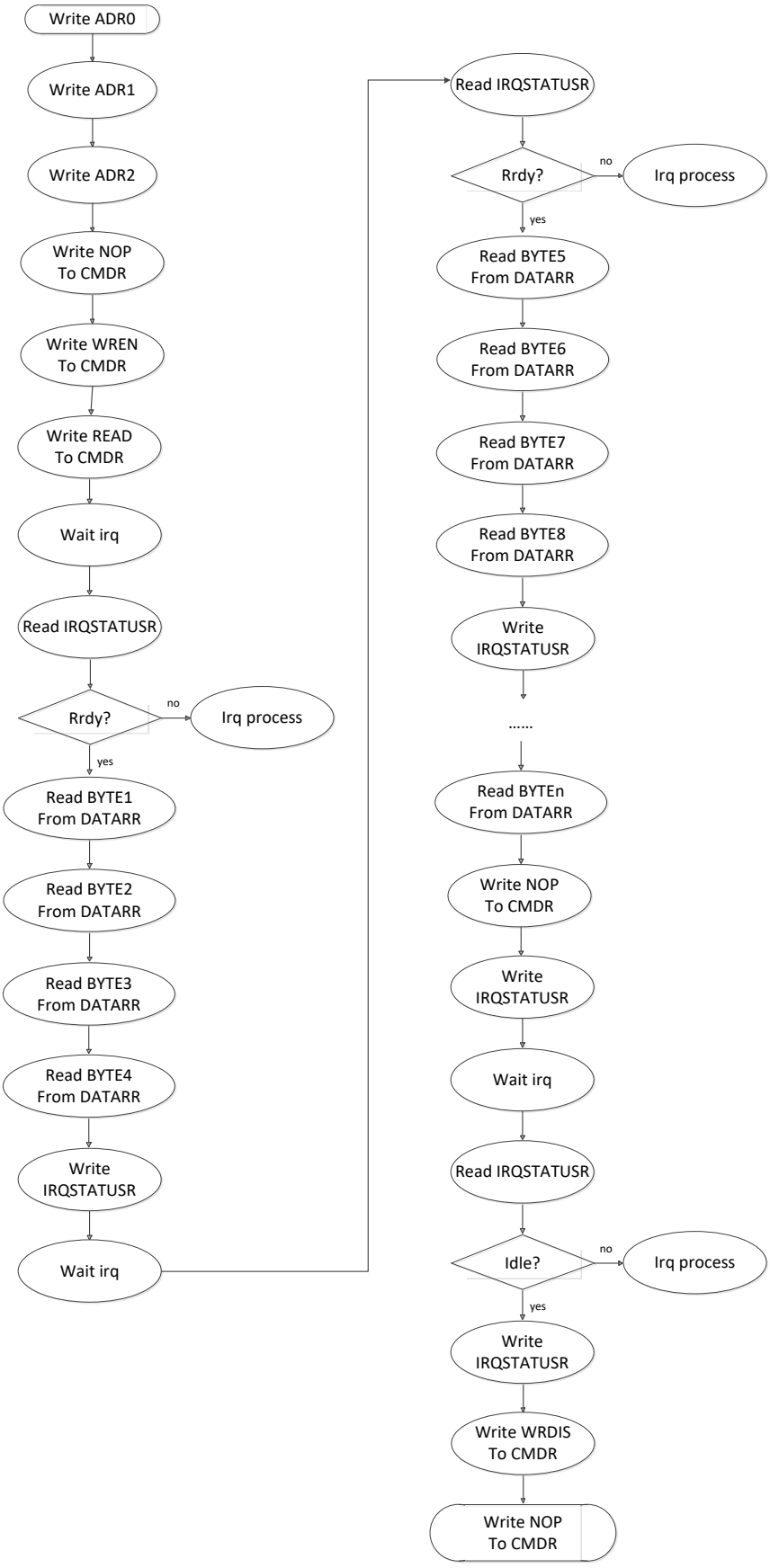

Figure 17-24 Read Bitstream

17.14 Program User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.14.1 Peer to Peer Mode

Table 17-22 Program User Flash

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the user Flash data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "PROGRAM" command to the command register
7	Write the first page of the user Flash data to the data transmit register
8	Continuously read the status register until 'busy' is detected low
9	Write the "READ" command to the command register
10	Read the first page of the user Flash data from the data receive register
11	Write the "NOP" command to the command register
12	Continuously read the status register until 'busy' is detected low
13	Write to address register 0
14	Write to address register 1
15	Write the address of the second page of the user Flash data to address register 2
16	Write the "NOP" command to the command register
17	Write the "PROGRAM" command to the command register
18	Write the second page of the user Flash data to the data transmit register
19	Continuously read the status register until 'busy' is detected low
20	Write the "READ" command to the command register
21	Read the second page of the user Flash data from the data receive
22	Write the "NOP" command to the command register
23	Continuously read the status register until 'busy' is detected low
24	……
25	Write to address register 0
26	Write to address register 1
27	Write the address of the nth page of the user Flash data to the address register 2
28	Write the "NOP" command to the command register
29	Write the "PROGRAM" command to the command register
30	Write the nth page of the user Flash data to the data transmit register
31	Continuously read the status register until 'busy' is detected low
32	Write the "READ" command to the command register

Steps	Operation Description
33	Read the nth page of the user Flash data from the data receive register
34	Write the "NOP" command to the command register
35	Continuously read the status register until 'busy' is detected low
36	Write the "WRDIS" command to the command register
37	Write the "NOP" command to the command register

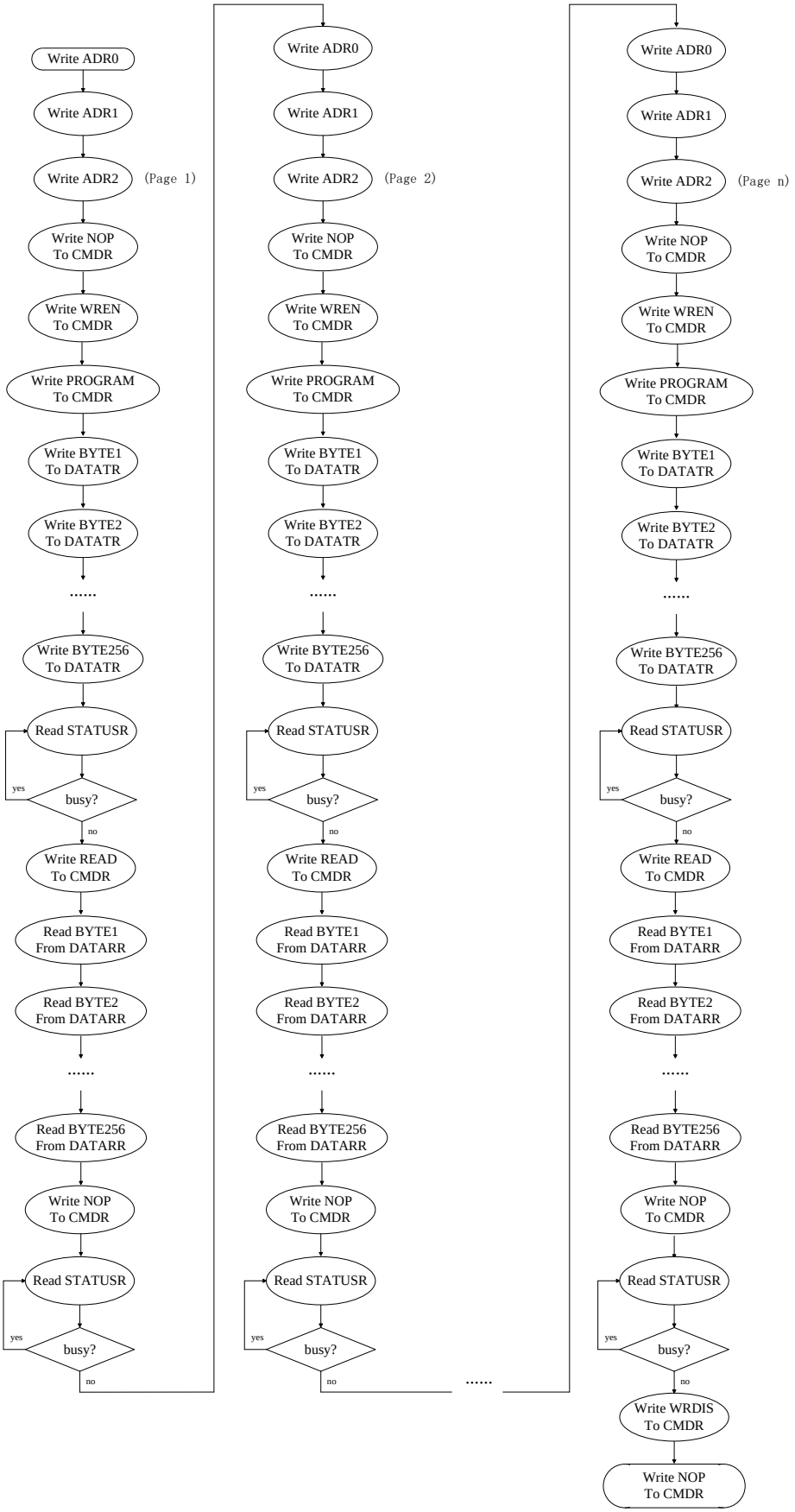

Figure 17-25 Program User Flash

17.14.2 Interrupt Mode

Table 17-23 Program User Flash

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the user Flash data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "PROGRAM" command to the command register
7	Write the first page of the user Flash data to the data transmit register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt Write the 1st byte to the data transmit register Write the 2nd byte to the data transmit register Write the 3rd byte to the data transmit register Write the 4th byte to the data transmit register Write to the interrupt status register to clear the embedded Flash write ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt Write the 5th byte to the data transmit register Write the 6th byte to the data transmit register Write the 7th byte to the data transmit register Write the 8th byte to the data transmit register Write to the interrupt status register to clear the embedded Flash write ready interrupt ..... After receiving an interrupt, read the interrupt status register to detect the embedded Flash write ready interrupt Write the 253rd byte to the data transmit register Write the 254th byte to the data transmit register Write the 255th byte to the data transmit register Write the 256th byte to the data transmit register Write to the interrupt status register to clear the embedded Flash write ready interrupt
8	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
9	Write to the interrupt status register to clear the embedded Flash idle interrupt
10	Write the "READ" command to the command register
11	Read the first page of the user Flash data from the data receive register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt

Steps	Operation Description
	Read the 253rd byte from the data receive register Read the 254th byte from the data receive register Read the 255th byte from the data receive register Read the 256th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt
12	Write the "NOP" command to the command register
13	Write to the interrupt status register to clear the embedded Flash read ready interrupt
14	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
15	Write to the interrupt status register to clear the embedded Flash idle interrupt
16	Write to address register 0
17	Write to address register 1
18	Write the address of the second page of the user Flash data to address register 2
19	Write the "PROGRAM" command to the command register
20	Write the second page of the user Flash data to the data transmit register
21	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
22	Write to the interrupt status register to clear the embedded Flash idle interrupt
23	Write the "READ" command to the command register
24	Read the second page of the user Flash data from the data receive
25	Write the "NOP" command to the command register
26	Write to the interrupt status register to clear the embedded Flash read ready interrupt
27	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
28	Write to the interrupt status register to clear the embedded Flash idle interrupt
29	……
30	Write to address register 0
31	Write to address register 1
32	Write the address of the nth page of the user Flash data to the address register 2
33	Write the "PROGRAM" command to the command register
34	Write the nth page of the user Flash data to the data transmit register
35	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
36	Write to the interrupt status register to clear the embedded Flash idle interrupt
37	Write the "READ" command to the command register
38	Read the nth page of the user Flash data from the data receive register
39	Write the "NOP" command to the command register
40	Write to the interrupt status register to clear the embedded Flash read ready interrupt
41	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
42	Write to the interrupt status register to clear the embedded Flash idle interrupt
43	Write the "WRDIS" command to the command register
44	Write the "NOP" command to the command register

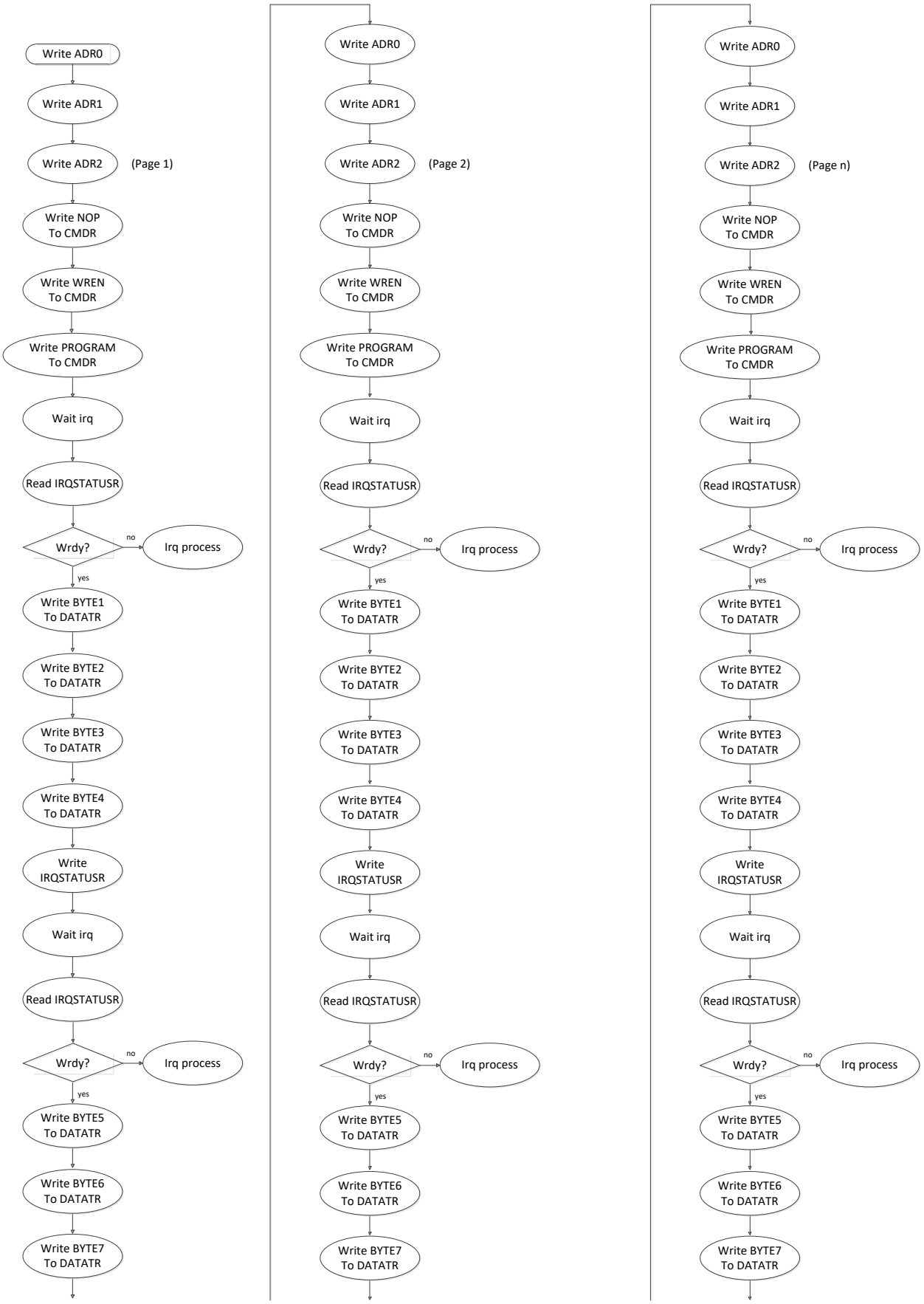

Figure 17-26 Program User Flash 1

Figure 17-27 Program User Flash 2

Figure 17-28 Program User Flash 3

17.15 Erase User Flash

17.15.1 Peer to Peer Mode

Table 17-24 Erase User Flash

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the user Flash data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "ERASE_PAGE" command to the command register
7	Continuously read the status register until 'busy' is detected low
8	Write the "READ" command to the command register
9	Read the first page of the user Flash data from the data receive register
10	Write the "NOP" command to the command register
11	Continuously read the status register until 'busy' is detected low
12	Write to address register 0
13	Write to address register 1
14	Write the address of the second page of the user Flash data to address register 2
15	Write the "NOP" command to the command register
16	Write the "ERASE_PAGE" command to the command register
17	Continuously read the status register until 'busy' is detected low
18	Write the "READ" command to the command register
19	Read the second page of the user Flash data from the data receive
20	Write the "NOP" command to the command register
21	Continuously read the status register until 'busy' is detected low
22	……
23	Write to address register 0
24	Write to address register 1
25	Write the address of the nth page of the user Flash data to the address register 2
26	Write the "NOP" command to the command register
27	Write the "ERASE_PAGE" command to the command register
28	Continuously read the status register until 'busy' is detected low
29	Write the "READ" command to the command register
30	Read the nth page of the user Flash data from the data receive register
31	Write the "NOP" command to the command register
32	Continuously read the status register until 'busy' is detected low
33	Write the "WRDIS" command to the command register
34	Write the "NOP" command to the command register

Figure 17-29 Erase User Flash

17.15.2 Interrupt Mode

Table 17-25 Erase User Flash

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the address of the first page of the user Flash data to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "ERASE_PAGE" command to the command register
7	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
8	Write to the interrupt status register to clear the embedded Flash idle interrupt
9	Write the "READ" command to the command register
10	Read the first page of the user Flash data from the data receive register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 253rd byte from the data receive register Read the 254th byte from the data receive register Read the 255th byte from the data receive register Read the 256th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt
11	Write the "NOP" command to the command register
12	Write to the interrupt status register to clear the embedded Flash read ready interrupt
13	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
14	Write to the interrupt status register to clear the embedded Flash idle interrupt
15	Write to address register 0
16	Write to address register 1
17	Write the address of the second page of the user Flash data to address register 2
18	Write the "ERASE_PAGE" command to the command register
19	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
20	Write to the interrupt status register to clear the embedded Flash idle interrupt
21	Write the "READ" command to the command register
22	Read the second page of the user Flash data from the data receive

Steps	Operation Description
23	Write the "NOP" command to the command register
24	Write to the interrupt status register to clear the embedded Flash read ready interrupt
25	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
26	Write to the interrupt status register to clear the embedded Flash idle interrupt
27	……
28	Write to address register 0
29	Write to address register 1
30	Write the address of the nth page of the user Flash data to the address register 2
31	Write the "ERASE_PAGE" command to the command register
32	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
33	Write to the interrupt status register to clear the embedded Flash idle interrupt
34	Write the "READ" command to the command register
35	Read the nth page of the user Flash data from the data receive register
36	Write the "NOP" command to the command register
37	Write to the interrupt status register to clear the embedded Flash read ready interrupt
38	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
39	Write to the interrupt status register to clear the embedded Flash idle interrupt
40	Write the "WRDIS" command to the command register
41	Write the "NOP" command to the command register

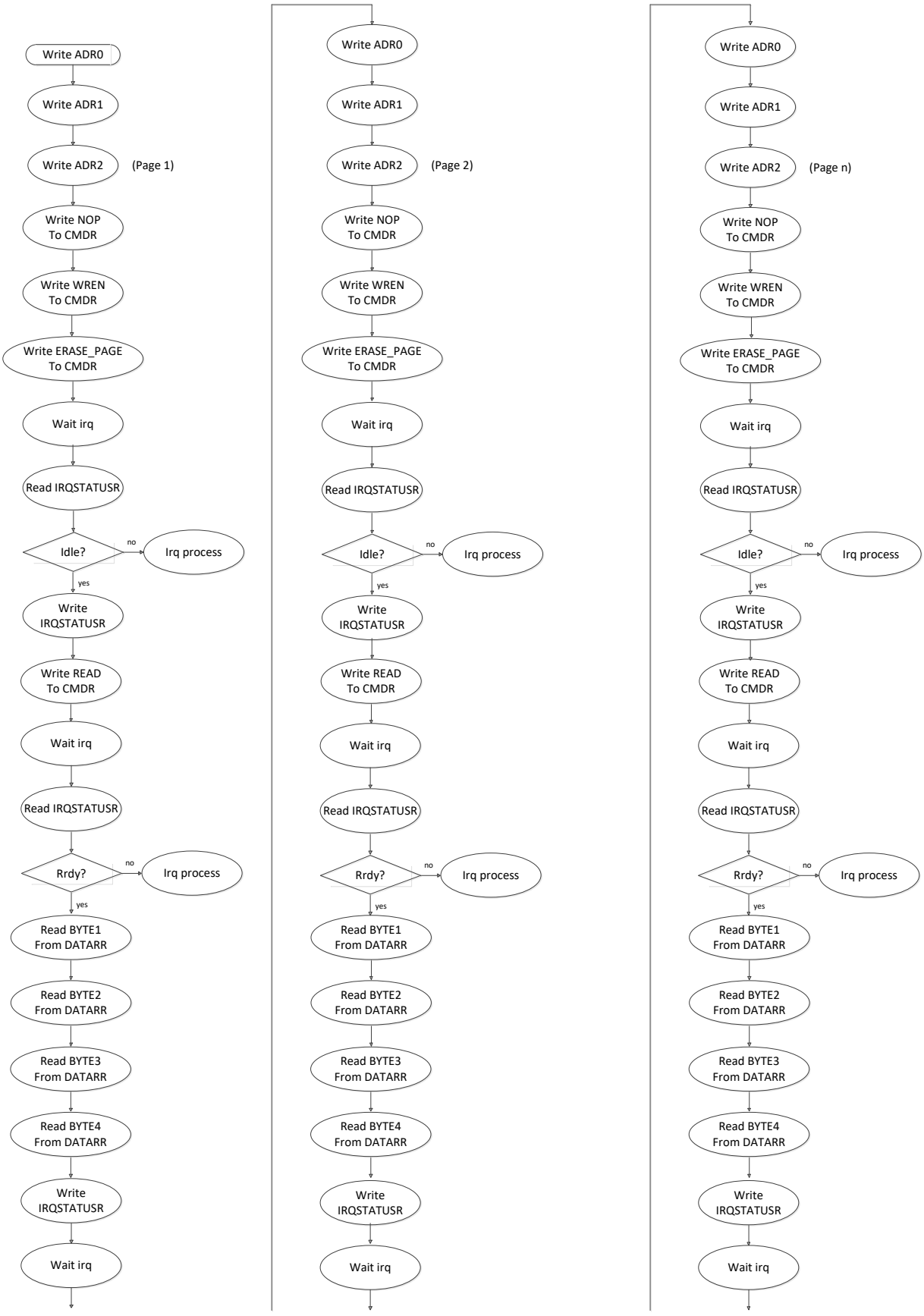

Figure 17-30 Erase User Flash 1

Figure 17-31 Erase User Flash 2

17.16 Erase Bitstream and User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.16.1 Peer to Peer Mode

Table 17-26 Erase Bitstream and User Flash

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "ERASE" command to the command register
4	Continuously read the status register until 'busy' is detected low
5	Write the "READ" command to the command register
6	Read bitstream data and user Flash data from the data receiver register
7	Write the "NOP" command to the command register
8	Continuously read the status register until 'busy' is detected low
9	Write the "WRDIS" command to the command register
10	Write the "NOP" command to the command register

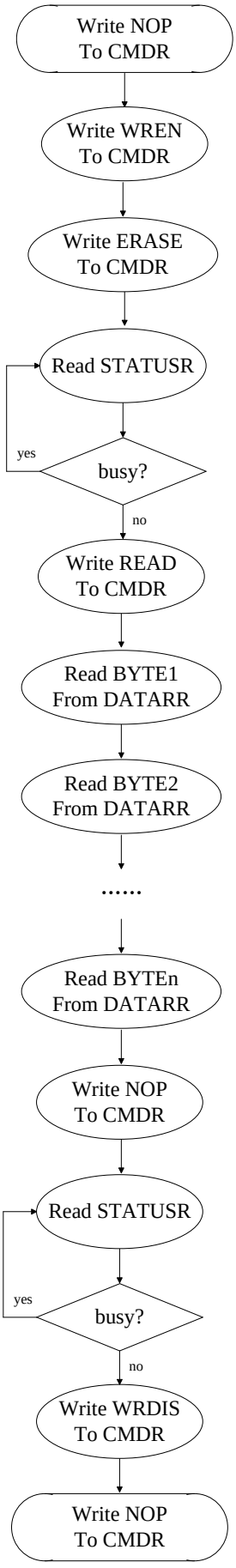

Figure 17-32 Erase Bitstream and User Flash

17.16.2 Interrupt Mode

Table 17-27 Erase Bitstream and User Flash

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the "ERASE" command to the command register
4	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
5	Write to the interrupt status register to clear the embedded Flash idle interrupt
6	Write the "READ" command to the command register
7	Read bitstream data and user Flash data from the data receiver register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... Read the nth byte from the data receiver register
8	Write the "NOP" command to the command register
9	Write to the interrupt status register to clear the embedded Flash read ready interrupt
10	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
11	Write to the interrupt status register to clear the embedded Flash idle interrupt
12	Write the "WRDIS" command to the command register
13	Write the "NOP" command to the command register

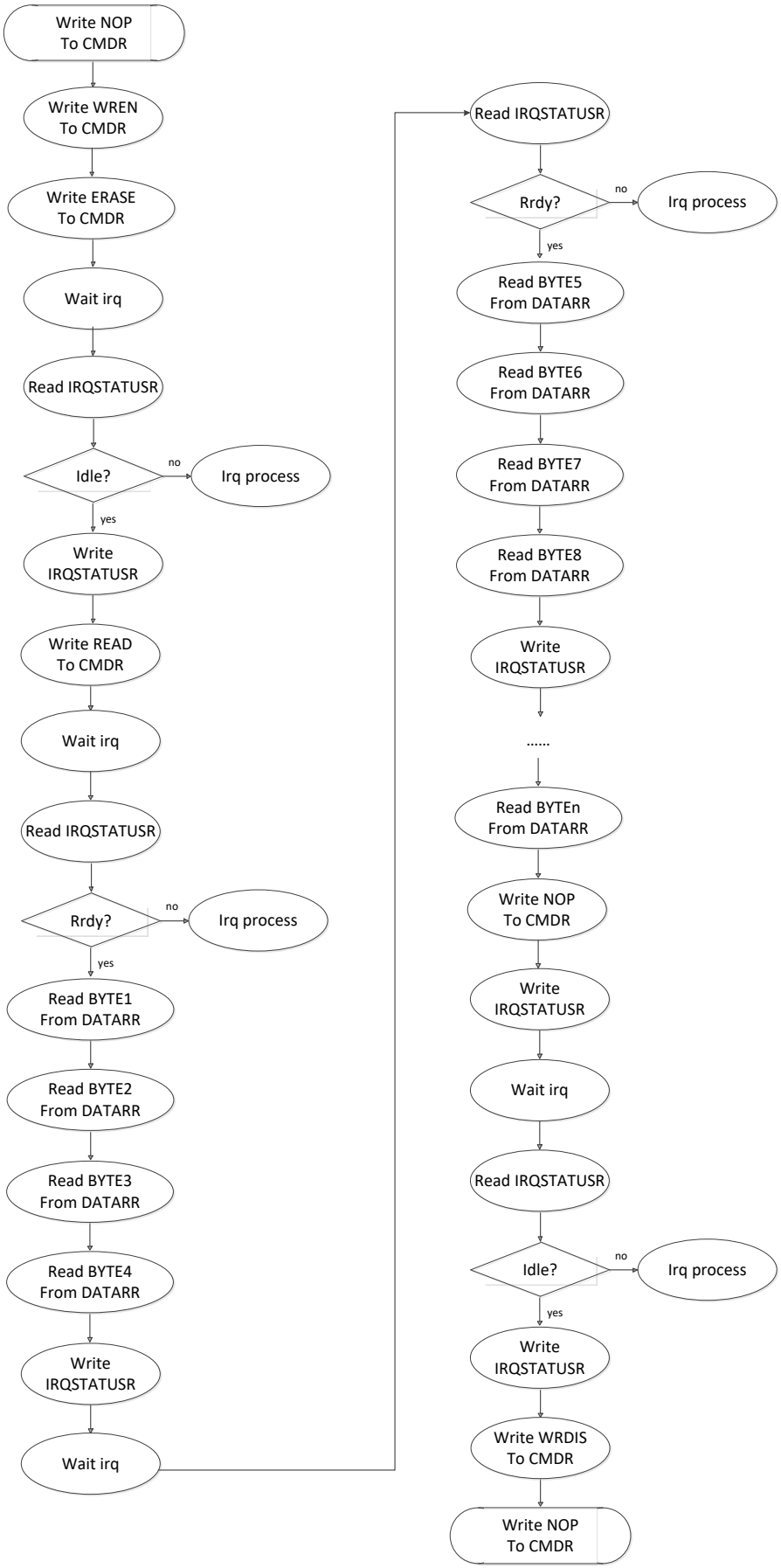

Figure 17-33 Erase Bitstream and User Flash

17.17 Read User Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.17.1 Peer to Peer Mode

Table 17-28 Read User Flash

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the starting address of the user Flash data in the embedded Flash to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "READ" command to the command register
7	Read user Flash data from the data receiver register
8	Write the "NOP" command to the command register
9	Continuously read the status register until 'busy' is detected low
10	Write the "WRDIS" command to the command register
11	Write the "NOP" command to the command register

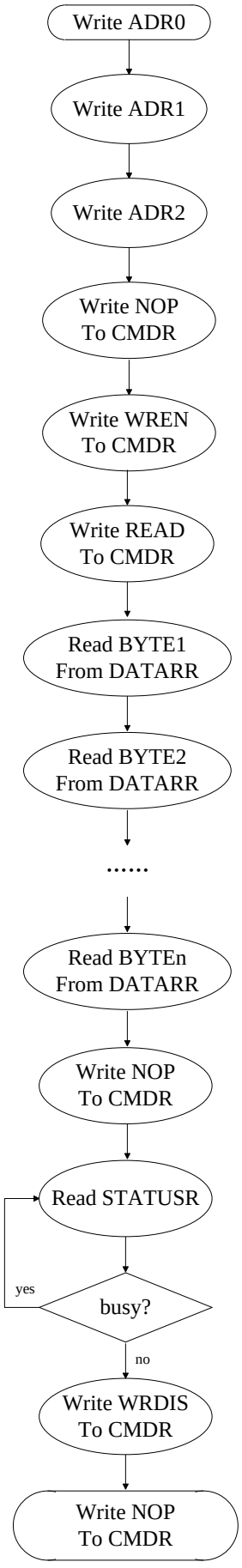

Figure 17-34 Read User Flash

17.17.2 Interrupt Mode

Table 17-29 Read User Flash

Steps	Operation Description
1	Write to address register 0
2	Write to address register 1
3	Write the starting address of the user Flash data in the embedded Flash to the address register 2
4	Write the "NOP" command to the command register
5	Write the "WREN" command to the command register
6	Write the "READ" command to the command register
7	Read user Flash data from the data receiver register: After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 1st byte from the data receive register Read the 2nd byte from the data receive register Read the 3rd byte from the data receive register Read the 4th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt After receiving an interrupt, read the interrupt status register to detect the embedded Flash read ready interrupt Read the 5th byte from the data receive register Read the 6th byte from the data receive register Read the 7th byte from the data receive register Read the 8th byte from the data receive register Write to the interrupt status register to clear the embedded Flash read ready interrupt ..... Read the nth byte from the data receiver register
8	Write the "NOP" command to the command register
9	Write to the interrupt status register to clear the embedded Flash read ready interrupt
10	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
11	Write to the interrupt status register to clear the embedded Flash idle interrupt
12	Write the "WRDIS" command to the command register
13	Write the "NOP" command to the command register

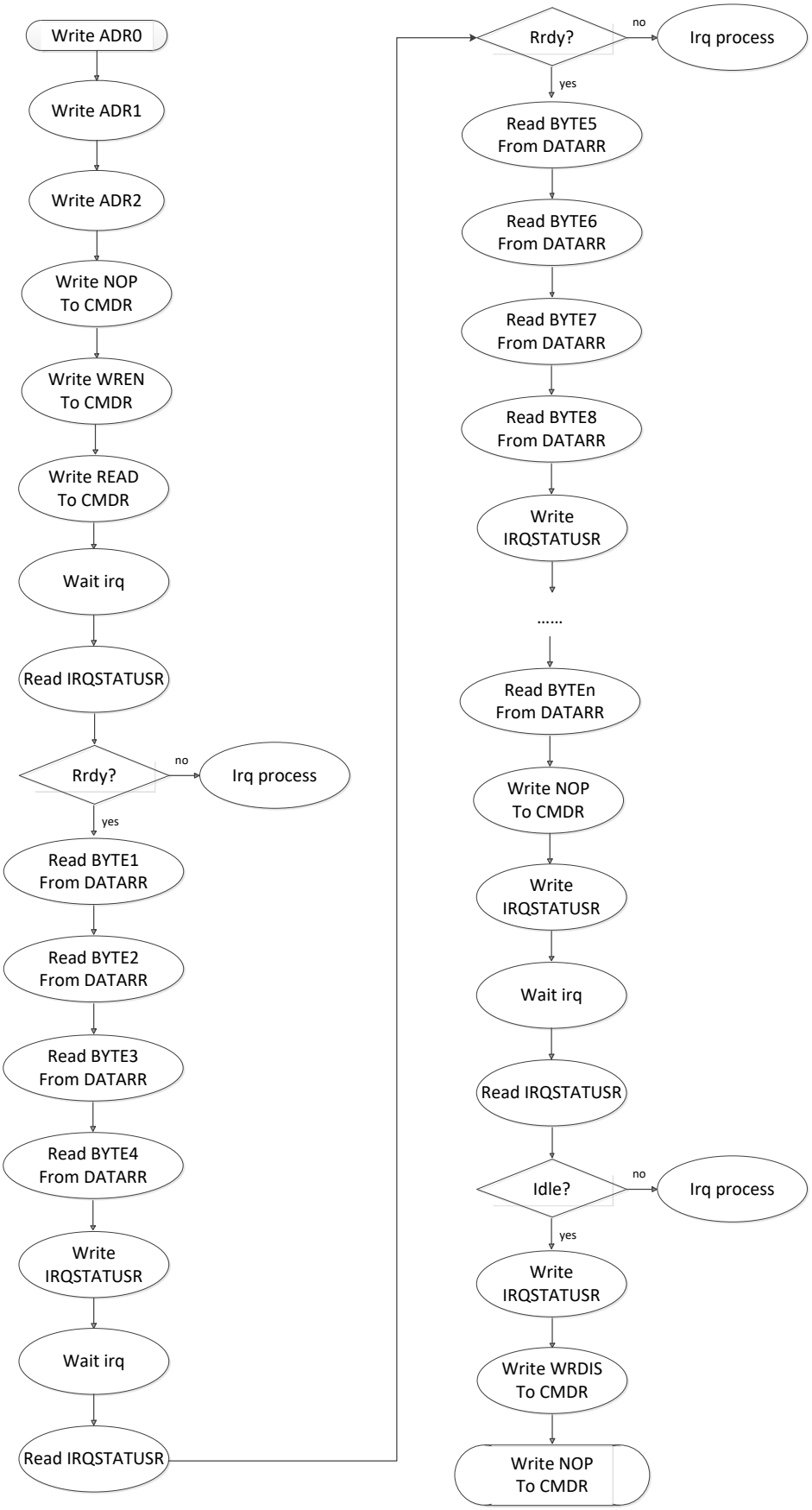

Figure 17-35 Read User Flash

17.18 Lock embedded Flash

First, wake up the embedded Flash. This operation can be initiated only when the embedded Flash is idle (the busy bit of the status register is 0).

17.18.1 Peer to Peer Mode

Table 17-30 Lock embedded Flash

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the “PROGRAM_LOCK” command to the command register
4	Continuously read the status register until 'busy' is detected low
5	Write the "WRDIS" command to the command register
6	Write the "NOP" command to the command register

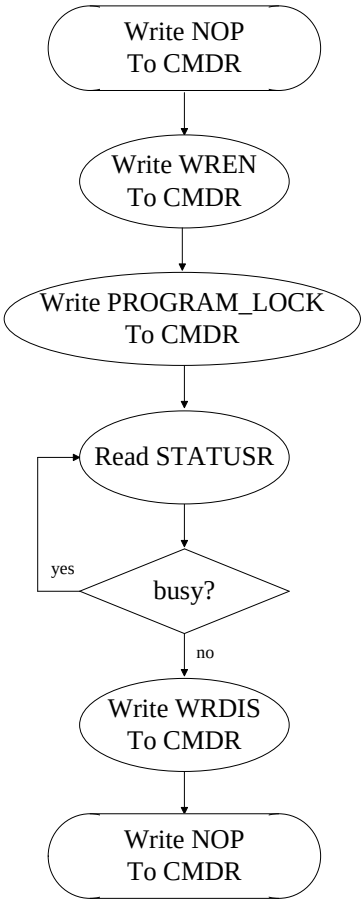

Figure 17-36 Lock embedded Flash

17.18.2 Interrupt Mode

Table 17-31 Lock embedded Flash

Steps	Operation Description
1	Write the "NOP" command to the command register
2	Write the "WREN" command to the command register
3	Write the “PROGRAM_LOCK” command to the command register
4	After receiving the interrupt, read the interrupt status register to detect the embedded Flash idle interrupt
5	Write to the interrupt status register to clear the embedded Flash idle interrupt
6	Write the "WRDIS" command to the command register
7	Write the "NOP" command to the command register

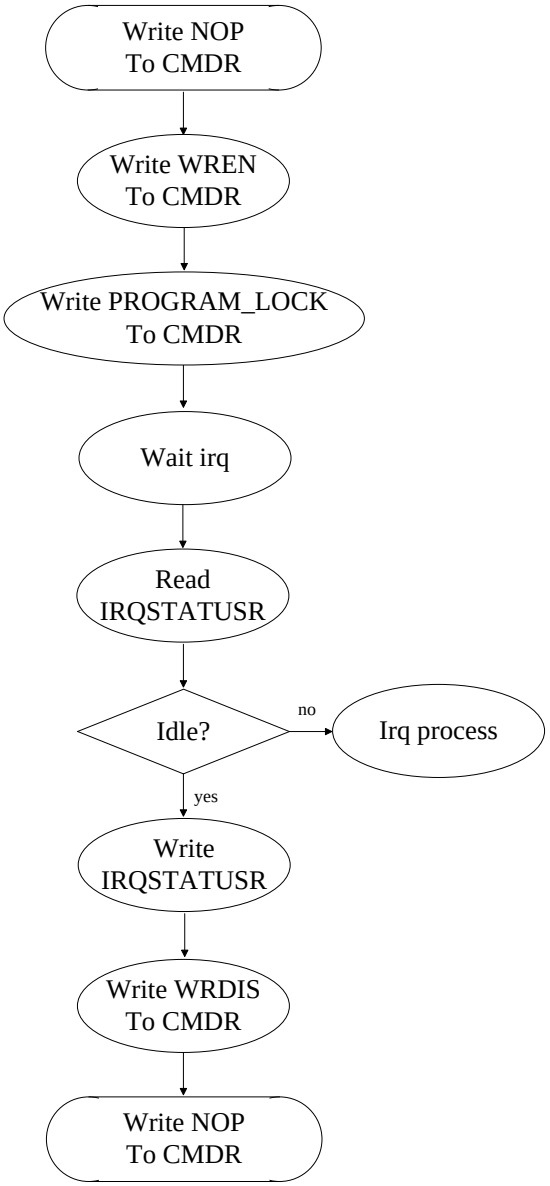

Figure 17-37 Lock embedded Flash

Chapter 18 Appendix 8

Description of TAP state machine.

18.1 Test-Logic-Reset

Test logic reset.

In the test logic reset state, all test logic is disabled, and the chip runs in operating mode.

When the TMS signal is 1, if the rising edges of TCK are more than five, the test logic reset state can be entered regardless of the state of the TAP controller.

Upon power-up of the chip, the TAP controller's initial state is the test logic reset state.

In the test logic reset state, the instruction register is initialised to the "IDCODE" instruction.

18.2 Run-Test/Idle

Test/Idle:

A state between scan operations. In the Test/Idle state, the test logic operation performed depends on the current instruction.

For some instructions, the Test/Idle state means waiting for the execution of the test instruction to be completed.

For some instructions, the Test/Idle state means no operation is required and the test logic remains idle.

18.3 Select-DR-Scan

Select data register scan.

18.4 Select-IR-Scan

Select instruction register scan.

18.5 Capture-DR

Data Capture.

In the Capture-DR state, when there is a rising edge on TCK, test data is loaded in parallel into the test data register specified by the current instruction.

If the test data register selected by the current instruction has no parallel input or there is no need to capture data for the selected test, the register will maintain its previous state.

18.6 Shift-DR

Data Shift.

In the Shift-DR state, when there is a rising edge on TCK, the test data register placed in the TDI to TDO scan chain shifts one bit towards TDO.

The test data register is selected by the current test instruction, and when it is not placed in the TDI to TDO scan chain, it will maintain its original state.

18.7 Exit1-DR

First Data Exit.

The Exit1-DR state is a transient process.

18.8 Pause-DR

Dada Pause.

In the Pause-DR state, the data shift of the test data register placed on the TDI to TDO link is paused.

18.9 Exit2-DR

Second Data Exit.

The Exit2-DR state is a transient process.

18.10 Update-DR

Data Update.

Some test data registers have latched parallel outputs that respond to specific test instructions. When data is shifted into the connected shift register path, it prevents changes at the parallel output end.

Upon entering the Update-DR state, the data from the shift register path is latched to the parallel output of the test data register on the falling edge of TCK.

In other states, the data latched to the parallel output end remains unchanged.

18.11 Capture-IR

Instruction Capture.

In the Capture-IR state, when there is a rising edge on TCK, the instruction register loads specially designed data.

18.12 Shift-IR

Instruction Shift.

In the Shift-IR state, the instruction register is placed on the TDI to TDO scan chain. When there is a rising edge on TCK, the instruction register shifts one bit towards TDO.

18.13 Exit1-IR

First Instruction Exit.

The Exit1-IR state is a transient process.

18.14 Pause-IR

Instruction Pause.

In the Pause-IR state, shifting of the instruction register is paused.

18.15 Exit2-IR

Second Instruction Exit.

The Exit2-IR state exit state is a transient process.

18.16 Update-IR

Instruction Update.

When entering the Update-IR state, the data from the shift register path is latched to the parallel output of the instruction register on the falling edge of TCK. Once the new instruction is latched, it becomes the current instruction.

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 series products at any time without prior notice.