



# **sysCONFIG Usage Guide for Nexus Platform**

## **Technical Note**

FPGA-TN-02099-1.2

March 2021

## Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

## Contents

|  |    |
|--|----|
| Acronyms in This Document .....  | 7  |
| 1. Introduction .....  | 8  |
| 2. Features .....  | 9  |
| 3. Definition of Terms .....   | 10 |
| 4. Configuration Details .....   | 11 |
| 4.1. Bitstream/PROM Sizes .....  | 11 |
| 4.2. sysCONFIG™ Ports .....  | 11 |
| 4.3. Configuration Ports Arbitration .....                                 | 12 |
| 4.4. sysCONFIG Pins .....  | 14 |
| 4.4.1. PROGRAMN .....  | 15 |
| 4.4.2. INITN .....   | 15 |
| 4.4.3. DONE .....  | 16 |
| 4.5. Master SPI sysCONFIG Pins .....                                       | 17 |
| 4.5.1. MCLK .....  | 17 |
| 4.5.2. MCSN .....  | 17 |
| 4.5.3. MOSI/MD0 .....  | 17 |
| 4.5.4. MISO/MD1 .....  | 18 |
| 4.5.5. MD2 .....   | 18 |
| 4.5.6. MD3 .....   | 18 |
| 4.5.7. MCSNO/MSDO .....  | 18 |
| 4.6. Slave sysCONFIG Pins .....  | 18 |
| 4.6.1. TCK/SCLK .....  | 18 |
| 4.6.2. TMS/SCSN .....  | 18 |
| 4.6.3. TDI/SI/SD0 .....  | 19 |
| 4.6.4. TDO/SD1 .....   | 19 |
| 4.6.5. SD2/SCL .....   | 19 |
| 4.6.6. SD3/SDA .....   | 19 |
| 4.7. PERSISTENT .....  | 19 |
| 5. Master Port Configuration Process and Flow .....                        | 20 |
| 5.1. Power-up Sequence .....   | 21 |
| 5.2. Initialization .....  | 21 |
| 5.3. Configuration .....   | 22 |
| 5.4. Wake-up .....   | 22 |
| 5.5. Early I/O Release .....   | 23 |
| 5.6. User Mode .....   | 23 |
| 5.7. Clearing the Configuration Memory and Re-initialization .....         | 23 |
| 5.8. Reconfiguration Priority .....  | 23 |
| 6. Configuration Modes .....   | 24 |
| 6.1. Master SPI Modes .....  | 24 |
| 6.1.1. Method to Enable the Master SPI Port .....                          | 25 |
| 6.1.2. Dual-Boot and Multi-Boot Configuration Modes .....                  | 25 |
| 6.1.3. Ping-Pong Boot .....  | 26 |
| 6.1.4. Dual and Quad Master SPI Read Mode .....                            | 27 |
| 6.2. Slave SPI Mode .....  | 29 |
| 6.2.1. Method to Enable the Slave SPI Port .....                           | 31 |
| 6.2.2. Specifications and Timing Diagrams – Slave SPI Port Waveforms ..... | 31 |
| 6.2.3. Slave SPI Port AC Timing Requirements .....                         | 32 |
| 6.2.4. Dual and Quad Slave SPI Port .....                                  | 32 |
| 6.2.5. Slave SPI Configuration Flow Diagrams .....                         | 33 |
| 6.2.6. Command Waveforms .....   | 34 |
| 6.2.7. Slave SPI to Master SPI Bridge .....                                | 36 |
| 6.3. Slave I <sup>2</sup> C/I <sup>3</sup> C Mode .....                    | 37 |

|        |   |    |
|--------|---|----|
| 6.3.1. | Bus Sharing Between I <sup>2</sup> C and I3C .....            | 38 |
| 6.3.2. | Slave I <sup>2</sup> C/I3C Configuration Mode .....           | 38 |
| 6.3.3. | Slave Address for I <sup>2</sup> C and I3C Ports .....        | 39 |
| 6.3.4. | Method to Enable the Slave I <sup>2</sup> C/I3C Port .....    | 39 |
| 6.3.5. | Slave I <sup>2</sup> C C/I3C Configuration Logic Access ..... | 40 |
| 6.3.6. | Slave I <sup>2</sup> C/I3C AC Timing Requirements .....       | 42 |
| 6.4.   | JTAG Mode .....   | 42 |
| 6.4.1. | Method to Enable the JTAG port .....                          | 42 |
| 6.4.2. | JTAG Port AC Timing Requirements .....                        | 43 |
| 6.4.3. | JTAG to Master SPI Bridge .....                               | 43 |
| 6.4.4. | JTAG ispTracy/Reveal Support .....                            | 43 |
| 6.5.   | Device Status Register .....                                  | 44 |
| 6.6.   | Control Register 0 (CR0) .....                                | 47 |
| 6.7.   | Control Register 1 (CR1) .....                                | 48 |
| 6.8.   | TransFR Operation .....                                       | 49 |
| 6.9.   | SPI/I <sup>2</sup> C/I3C Command Support .....                | 49 |
| 6.10.  | JTAG Instruction Support .....                                | 52 |
| 7.     | Software Selectable Options .....                             | 59 |
| 7.1.   | SLAVE_SPI_PORT .....  | 60 |
| 7.2.   | MASTER_SPI_PORT .....   | 61 |
| 7.3.   | SLAVE_I2CI3C_PORT .....                                       | 61 |
| 7.4.   | JTAG_PORT .....   | 61 |
| 7.5.   | DONE_PORT .....   | 61 |
| 7.6.   | INITN_PORT .....  | 62 |
| 7.7.   | PROGRAMN_PORT .....   | 62 |
| 7.8.   | BACKGROUND_RECONFIG .....                                     | 62 |
| 7.9.   | DONE_EX .....   | 62 |
| 7.10.  | DONE_OD .....   | 62 |
| 7.11.  | MCCLK Frequency .....   | 63 |
| 7.12.  | TRANSFR .....   | 63 |
| 7.13.  | CONFIG_IOVOLTAGE .....  | 63 |
| 7.14.  | CONFIG_IOSLEW .....   | 63 |
| 7.15.  | CONFIG_SECURE .....   | 63 |
| 7.16.  | WAKE_UP .....   | 63 |
| 7.17.  | COMPRESS_CONFIG .....   | 64 |
| 7.18.  | EARLY_IO_RELEASE .....  | 64 |
| 7.19.  | BOOTMODE .....  | 64 |
| 7.20.  | USERCODE_FORMAT .....   | 64 |
| 7.21.  | USERCODE .....  | 64 |
| 7.22.  | UNIQUE_ID .....   | 64 |
| 8.     | Device Wake-up Sequence .....                                 | 65 |
| 8.1.   | Wake-up Signals .....   | 65 |
| 8.2.   | Wake-up Clock .....   | 66 |
| 9.     | Daisy Chaining .....  | 67 |
| 9.1.   | Bypass Option .....   | 67 |
| 9.2.   | Flow Through Option .....                                     | 68 |
|        | Appendix A. Nexus Slave SPI Programming Guide .....           | 70 |
|        | Appendix B. Nexus Bitstream File Format .....                 | 71 |
|        | Configuration Bitstream Format .....                          | 71 |
|        | Read Back Bitstream Format .....                              | 84 |
|        | Nexus Device Specific .....                                   | 84 |
|        | Technical Support Assistance .....                            | 85 |
|        | Revision History .....  | 86 |

## Figures

|   |    |
|---|----|
| Figure 4.1. Configuration Control Flow.....   | 13 |
| Figure 4.2. Configuration from PROGRAMN Timing.....   | 15 |
| Figure 4.3. Configuration Error Notification .....  | 16 |
| Figure 5.1. Master Port Configuration Flow .....  | 20 |
| Figure 5.2. Configuration from Power-On-Reset Timing .....  | 21 |
| Figure 6.1. Nexus Master SPI Port with SPI Flash .....  | 25 |
| Figure 6.2. Jump Table.....   | 27 |
| Figure 6.3. One Dual SPI Flash Interface (Dual) .....   | 28 |
| Figure 6.4. One Quad SPI Flash Interface (Quad) .....   | 28 |
| Figure 6.5. Nexus Slave SPI Port with CPU and Single or Multiple Devices .....                                | 30 |
| Figure 6.6. Nexus Slave SPI Port with SPI Flash .....   | 30 |
| Figure 6.7. Slave SPI Read Waveforms .....  | 31 |
| Figure 6.8. Slave SPI Write Waveforms .....   | 32 |
| Figure 6.9. Slave SPI Configuration Flow .....  | 33 |
| Figure 6.10. Class A Command Waveforms.....   | 34 |
| Figure 6.11. Class B Command Waveforms.....   | 34 |
| Figure 6.12. Class C Command Waveforms.....   | 35 |
| Figure 6.13. Class D Command Waveforms.....   | 35 |
| Figure 6.14. SSPI to MSPI Bridge Functional Diagram .....   | 36 |
| Figure 6.15. SSPI to MSPI Bridge Block Diagram.....   | 36 |
| Figure 6.16. Nexus Slave I <sup>2</sup> C/I <sup>3</sup> C Port with CPU and Single or Multiple Devices ..... | 37 |
| Figure 6.17. Bus Sharing between I <sup>2</sup> C and I <sup>3</sup> C.....                                   | 38 |
| Figure 6.18. Slave I <sup>2</sup> C Configuration Activation Flow.....  | 40 |
| Figure 6.19. Slave I <sup>3</sup> C Configuration Activation Flow .....                                       | 40 |
| Figure 6.20. Typical I <sup>2</sup> C Write.....  | 40 |
| Figure 6.21. Typical I <sup>2</sup> C Read.....   | 40 |
| Figure 6.22. Typical I <sup>3</sup> C Write with 7'h7E.....   | 41 |
| Figure 6.23. Typical I <sup>3</sup> C Write without 7'h7E .....   | 41 |
| Figure 6.24. Typical I <sup>3</sup> C Read with 7'h7E.....  | 41 |
| Figure 6.25. Typical I <sup>3</sup> C Read without 7'h7E .....  | 41 |
| Figure 6.26. JTAG to MSPI Bridge Block Diagram .....  | 43 |
| Figure 6.27. JTAG ispTracy Interface.....   | 44 |
| Figure 7.1. sysCONFIG Preferences in Global Tab, Lattice Radiant Device Constraint Editor .....               | 59 |
| Figure 8.1. Wake-up Sequence Using Internal Clock.....  | 65 |
| Figure 9.1. Nexus in Configuration Daisy Chain with Master SPI in Bypass Mode .....                           | 67 |
| Figure 9.2. Nexus in Configuration Daisy Chain with Slave SPI in Bypass Mode .....                            | 68 |
| Figure 9.3. Nexus in Configuration Daisy Chain with Master SPI in Flow Through Mode .....                     | 68 |
| Figure 9.4. Nexus in Configuration Daisy Chain with Slave SPI in Flow Through Mode .....                      | 69 |

## Tables

|   |    |
|---|----|
| Table 4.1. Maximum Configuration Bits .....   | 11 |
| Table 4.2. Nexus Programming and Configuration Ports.....                               | 11 |
| Table 4.3. Slave Configuration Port Activation Key .....                                | 12 |
| Table 4.4. Default State of the sysCONFIG Pins .....                                    | 14 |
| Table 4.5. Nexus MCLK Valid Frequencies .....   | 17 |
| Table 4.6. sysCONFIG Pins Global Preferences .....                                      | 19 |
| Table 6.1. Master SPI Configuration Port Pins .....                                     | 24 |
| Table 6.2. Dual/Quad SPI Port Names .....   | 28 |
| Table 6.3. Slave SPI Configuration Port Pins .....                                      | 29 |
| Table 6.4. Slave SPI Configuration Port Activation Key.....                             | 31 |
| Table 6.5. Slave SPI port AC Timing Requirements .....                                  | 32 |
| Table 6.6. Special Commands for Dual and Quad Mode Enable/Disable .....                 | 32 |
| Table 6.7. Slave I <sup>2</sup> C/I <sup>3</sup> C Configuration Port Pins .....        | 37 |
| Table 6.8. Slave SPI Configuration Port Activation Key.....                             | 39 |
| Table 6.9. Slave I <sup>2</sup> C/I <sup>3</sup> C Maximum Frequency Requirements ..... | 42 |
| Table 6.10. JTAG AC Timing Requirements .....   | 43 |
| Table 6.11. 32-Bit Device Status Register .....   | 45 |
| Table 6.12. Bit Definition for Control Register 0.....                                  | 47 |
| Table 6.13. Bit Definition for Control Register 1.....                                  | 48 |
| Table 6.14. Slave Configuration Interface Command Table .....                           | 49 |
| Table 6.15. JTAG Instruction Table .....  | 52 |
| Table 7.1. sysCONFIG Options .....  | 60 |
| Table 8.1. Wake-up Sequences.....   | 65 |
| Table B.1. Nexus Compressed Bitstream Format .....                                      | 71 |
| Table B.2. Nexus Uncompressed Bitstream Format – Without Early I/O Release .....        | 76 |
| Table B.3. Nexus Uncompressed Bitstream Format – With Early I/O Release .....           | 80 |
| Table B.4. Nexus Read Back File Format.....   | 84 |
| Table B.5. Nexus Device Specifics .....   | 84 |
| Table B.6. Nexus Device ID .....  | 84 |

## Acronyms in This Document

A list of acronyms used in this document.

| Acronym | Definition  |
|---------|---|
| AES     | Advanced Encryption Standard  |
| CRC     | Cyclic Redundancy Check   |
| EBR     | Embedded Block RAM  |
| ECDSA   | Elliptic Curve Digital Signature Algorithm  |
| HMAC    | Hash-based Message Authentication Code  |
| I2C     | Inter-Integrated Circuit; A synchronous, multi-master, multi-slave, packet switched, single-ended, serial bus |
| JTAG    | Joint Test Action Group   |
| LMMI    | Lattice Memory Mapped Interface   |
| LSB     | Least Significant Bit   |
| LUT     | Look Up Table   |
| MSB     | Most Significant Bit  |
| MSPI    | Master Serial Peripheral Interface  |
| OTP     | One Time Programmable   |
| PCB     | Printed Circuit Board   |
| POR     | Power On Reset  |
| SCM     | Serial Configuration Mode   |
| SEC     | Soft Error Correction   |
| SED     | Soft Error Detection  |
| SER     | Soft Error Rate   |
| SPI     | Serial Peripheral Interface   |
| SRAM    | Static Random-Access Memory   |
| SSPI    | Slave Serial Peripheral Interface   |
| TCK     | Test Clock Pin  |
| TDI     | Test Data Input   |
| TDO     | Test Data Output  |
| TMS     | Test Mode Select  |

# 1. Introduction

The Lattice Nexus™ Platform is the next generation of Lattice FPGAs based on the FDSOI technology, which includes CrossLink™-NX and Certus™-NX product families. These devices reap many benefits of FDSOI technology like extremely low soft error rate (SER) and easy configurability of the same device for high performance as well as low power applications. The Nexus device has many new and unique features that make it one of the best in class FPGA in its device density range. The Nexus device has one of the fastest boot up times including ultrafast I/O booting capability, which moves this device a notch higher for booting compared to traditional FPGAs. This device has advanced options to enable multi-boot feature so you can easily switch between FPGA bitstreams.

The configuration memory in Nexus FPGA is built using volatile SRAM; therefore, an external non-volatile configuration memory or external configuration engine is required to maintain the configuration data when the power is removed. This non-volatile memory or configuration engine supplies the configuration data to the Nexus device when it powers up or anytime the device needs to be updated. The Nexus device provides a rich set of features for configuring the FPGA or programming the external non-volatile memory. You have many options available to you for building the programming solution that fits your needs. Each of the options available are described in detail so that you can put together the programming and configuration solution that meets your needs. Waveforms presented in this document are for reference only; for detailed timing recommendations, refer to [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).



## 2. Features

Key programming and configuration features of Nexus device are:

- Ultrafast I/O configuration for instant-on support
- Fast full device configuration using quad SPI running at 150 MHz
- I3C device configuration support
- Bitstream dry-run support to ensure bitstream integrity
- Extremely low Soft Error Rate (SER) due to inherent FDSOI technology
- Easy switching between the high performance and low power modes using different bitstream configurations
- Enhanced and flexible Multi-boot support (32-bit addressing support with jump-table support)
- Configuration bridging for easy external SPI programming (SSPI to Master SPI bridge)
- User-selectable Booting Sequence
- Bitstream Encryption/Decryption – 256 bits AES (Available for Lattice Radiant™ software V2.0SP1 or later). For detailed information about the cypher key handling and advanced security feature, refer to [Advanced Configuration Security Usage Guide for Nexus Platform \(FPGA-TN-02176\)](#).
- Bitstream Authentication – HMAC (Available for Lattice Radiant® Software V2.2 or later), for Certus-NX product family. Refer to [Advanced Configuration Security Usage Guide for Nexus Platform \(FPGA-TN-02176\)](#) for detailed information.
- Bitstream Authentication – ECDSA (Available for Lattice Radiant® Software V2.2 or later), for Certus-NX product family. Refer to [Advanced Configuration Security Usage Guide for Nexus Platform \(FPGA-TN-02176\)](#) for detailed information.
- Multiple programming and configuration interfaces:
  - 1149.1 JTAG
  - Slave I<sup>2</sup>C
  - Slave I3C
  - Slave SPI, includes SERIAL, DUAL, and QUAD mode
  - Master SPI includes SERIAL, DUAL, and QUAD read mode
  - Dual Boot and Multi Boot support
  - Ping-Pong Boot
  - Daisy chaining
- Transparent programming of external memory
- Readback security and encryption for design protection
- Compression of bitstreams

### 3. Definition of Terms

This document uses the following terms to describe common functions:

- **Programming** – Programming refers to the process used to alter the contents of the external configuration memory.
- **Configuration** – Configuration refers to a change in the state of the SRAM memory cells.
- **Configuration Mode** – The configuration mode defines the method the device uses to acquire the configuration data from the volatile memory.
- **Configuration Data** – This is the data read from the non-volatile memory and loaded into the FPGA's SRAM configuration memory. This is also referred to as a bitstream, or device bitstream.
- **BIT** – The BIT file is the configuration data for the device that is stored in an external SPI Flash or other memory device. It is a binary file and is programmed unmodified into the SPI Flash by Lattice programming tool.
- **HEX** – The HEX file is also a configuration data file for the device. It is in HEX format and is normally requested by third party programming vendors.
- **Port** – A port refers to the physical connection used to perform programming and some configuration operations. Ports on the Nexus device include JTAG and SPI physical connections.
- **User Mode** – The Nexus device is in user mode when configuration is complete, and the FPGA is performing the logic functions you have programmed it to perform.
- **Offline Mode** – Offline mode is a term that is applied to SRAM configuration or external memory programming. When using offline mode, the FPGA no longer operates in user mode. The contents of the SRAM configuration memory or external memory device are updated. The FPGA does not perform your logic operations until offline mode programming/configuration is complete.
- **Direct Mode (Foreground Mode)** – The device is in a configuration mode and all the I/O pins are kept tristated.
- **Dual Boot** – Supports two configuration patterns that reside in a SPI Flash device. Whenever loading failure occurs with the primary pattern, the FPGA device searches for and loads a so-called *golden* or fail-safe pattern. Both patterns come from an off-chip non-volatile SPI memory.
- **Ping-Pong Boot** – Utilize the Jump Table to select an image for booting without changing location of the image in SPI flash.
- **Multi Boot** – The FPGA device determines and triggers the loading of the next pattern after a prior successful configuration. Multiple patterns (that is, more than two patterns) are available for the FPGA device to choose to load on demand. All the patterns are stored in an external SPI Flash memory.
- **Transparent Mode** – Transparent mode, also referred to as Background Mode, is used to update the SRAM configuration while leaving the Nexus device in user mode and all I/O pins remain operational.
- **Number Formats** – The following nomenclature is used to denote the radix of numbers
  - **0x**: Numbers preceded by *0x* are hexadecimal
  - **b (suffix)**: Numbers suffixed with *b* are binary
  - All other numbers are decimal
- **SPI** – Serial Peripheral Interface Bus. An industry standard, full duplex, synchronous serial data link that uses a four-wire interface. The interface supports a single master and single or multiple slaves.
- **Master SPI** – A configuration mode where the FPGA drives the master clock and issues commands to read the bitstream from an external SPI Flash device.
- **Slave SPI (SSPI)** – A configuration mode where the CPU drives the clock and issues commands to the FPGA for writing the bitstream into the SRAM cells.
- **Refresh** – The process of re-triggering a bitstream write operation. It is activated by toggling the PROGRAMN pin or issuing a REFRESH command, which emulates the PROGRAMN pin toggling. Only the JTAG port and the Slave SPI port support the REFRESH command.
- **Dry-Run** – The process triggered by the LSC\_DEVICE\_CONTROL command, which loads the bitstream and checks the CRC of the non-volatile bits without actually writing the bits to the configuration SRAM (that is, it is done in the background during normal device operation), for the purpose of checking the bitstream file integrity.

## 4. Configuration Details

The Nexus SRAM memory contains the active configuration, essentially the *fuses* that define the behavior of the FPGA. The active configuration is, in most cases, retrieved from an external memory. The non-volatile memory holds the configuration data that is loaded into the FPGAs SRAM.

### 4.1. Bitstream/PROM Sizes

The Nexus devices are SRAM based FPGAs. The SRAM configuration memory must be loaded from an external non-volatile memory that can store all of the configuration data. The size of the configuration data is variable. It is based on the amount of logic available in the FPGA, and the number of pre-initialized Embedded Block RAM (EBR) components and number of pre-initialized Large RAM block (LRAM) components. A Nexus design using the largest device, with every EBR and every LRAM pre-initialized with unique data values, and generated without compression turned on requires the largest amount of storage.

**Table 4.1. Maximum Configuration Bits**

| Device                | Scenario          | All Uncompressed                          | SPI Mode                        |   |
|-----------------------|-------------------|---|---------------------------------|---|
|                       |                   | Unencrypted/Encrypted Bitstream Size (Mb) | Recommended SPI Flash Size (Mb) | Dual Boot Recommended SPI Flash Size (Mb) |
| LIFCL-17<br>LFD2NX-17 | No LRAM, No EBR,  | 2.817                                     | 4                               | 8   |
|                       | No LRAM, MAX EBR  | 3.273                                     | 4                               | 8   |
|                       | MAX LRAM, No EBR  | 5.517                                     | 8                               | 16  |
|                       | MAX LRAM, MAX EBR | 5.873                                     | 8                               | 16  |
| LIFCL-40<br>LFD2NX-40 | No LRAM, No EBR,  | 6.232                                     | 8                               | 16  |
|                       | No LRAM, MAX EBR  | 7.758                                     | 8                               | 16  |
|                       | MAX LRAM, No EBR  | 7.281                                     | 8                               | 16  |
|                       | MAX LRAM, MAX EBR | 8.807                                     | 16                              | 32  |

**Note:** Both unencrypted and encrypted bitstreams are the same size. Compression ratio depends on bitstream, so we only provide uncompressed bitstream data.

### 4.2. sysCONFIG™ Ports

**Table 4.2. Nexus Programming and Configuration Ports**

| Interface | Port                                       | Description                                    |
|-----------|--|--|
| JTAG      | JTAG (IEEE 1149.1 and IEEE 1532 compliant) | 4-wire or 5-wire JTAG Interface                |
| sysCONFIG | SSPI (SERIAL, DUAL, QUAD)                  | Slave Serial Peripheral Interface (SPI)        |
|           | MSPI (SERIAL, DUAL, QUAD)                  | Master Serial Peripheral Interface (SPI)       |
|           | Slave I <sup>2</sup> C                     | Slave I <sup>2</sup> C configuration interface |
|           | Slave I3C                                  | Slave I3C configuration interface              |

### 4.3. Configuration Ports Arbitration

At Power Up or PROGRAMN pin toggle LOW (for the period of tPROGRAMN) or REFRESH command execution, the configuration logic puts the device into master auto booting mode to boot the device from external SPI boot PROM, if the PROGRAMN pin is *HIGH*, after a brief internal initialization time (Bulk erase the Configuration SRAM and CDM downloading). Holding the PROGRAMN LOW postpones the master auto-booting event, and allow the slave configuration ports (Slave SPI or Slave I<sup>2</sup>C) to detect a *Slave Active* condition, which means: slave port is addressed, and the pre-defined Slave Configuration Port Activation Key, as shown in [Table 4.3](#) is received. If any slave port declares active before PROGRAMN is released HIGH, the device is activated for slave configuration. If no slave port is declared active before the PROGRAMN pin is released HIGH, the device performs master auto booting sequence.

The PROGRAMN pin becomes a user I/O pin via a user feature setting, with default PROGRAMN pin state high to achieve a Master booting only device. The PROGRAMN pin state could be set low to achieve a slave port configuration only device.

The data format for slave configuration port activation is shown below

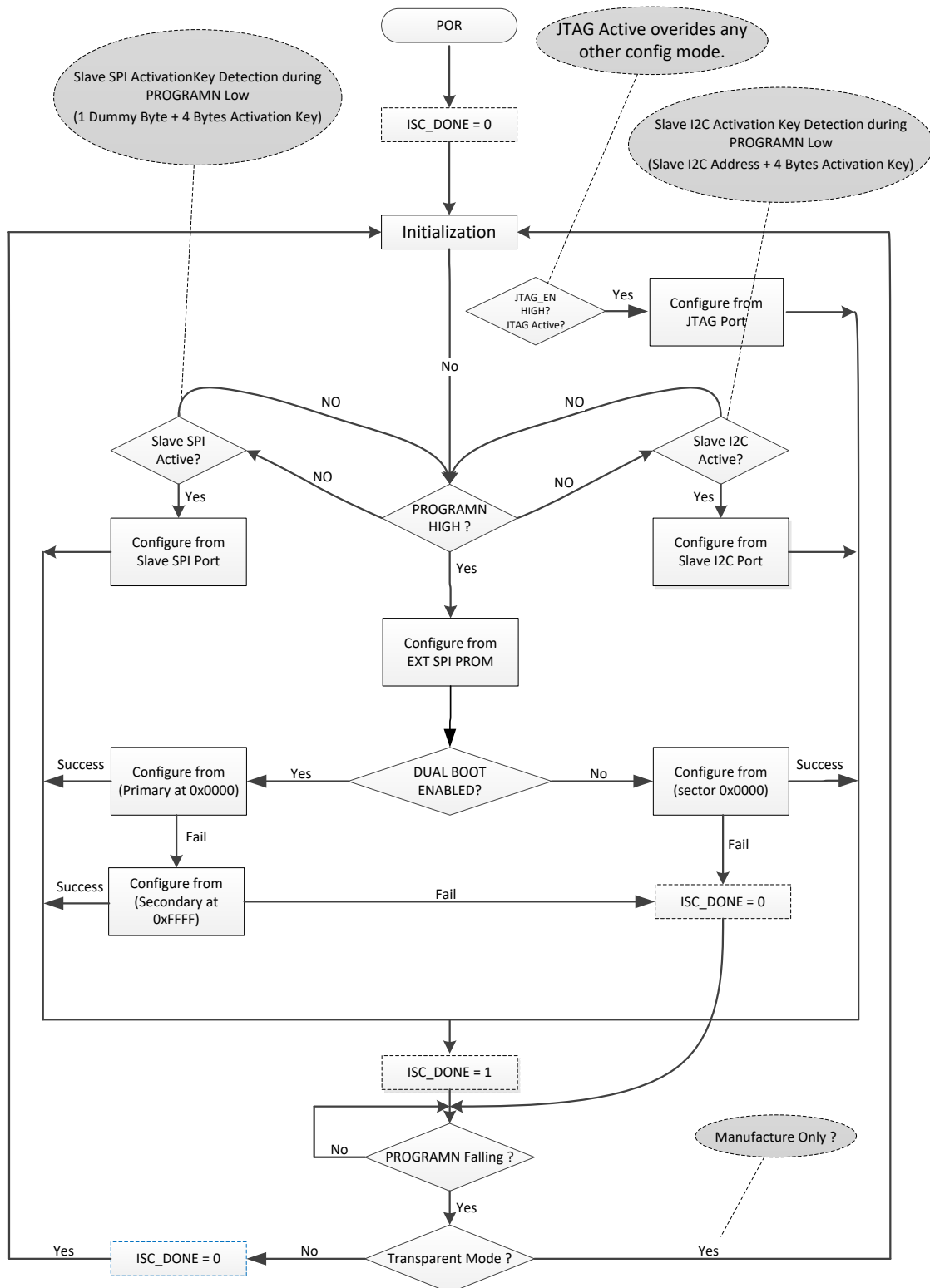
**Table 4.3. Slave Configuration Port Activation Key**

| Slave Port / Activation Key                  | Slave Configuration Port Activation Key                           |              |
|--|---|--------------|
| Slave SPI Port                               | Dummy Bytes <sup>1</sup>  | 32'HA4C6F48A |
| Slave I <sup>2</sup> C/I <sup>3</sup> C Port | Slave I <sup>2</sup> C/I <sup>3</sup> C Port Address <sup>2</sup> | 32'HA4C6F48A |

**Notes:**

1. The number of dummy bytes should be at least 1, the last shifted in 32 bits matter.
2. The slave I<sup>2</sup>C address could be either 7 bits or 10 bits address.

The device configuration control flow of CFG\_TOP is shown in [Figure 4.1](#).



**Figure 4.1. Configuration Control Flow**

JTAG has higher priority than other modes. If JTAG becomes active by driving JTAG\_ENABLE HIGH during any other boot mode, that mode is canceled and JTAG takes over.

## 4.4. sysCONFIG Pins

The Nexus device provides a set of sysCONFIG I/O pins that you use to program and configure the FPGA. The sysCONFIG pins are grouped together to create ports (such as JTAG, SSPI, MSPI) that are used to interact with the FPGA for programming, configuration, and access of resources inside the FPGA. The sysCONFIG pins in a configuration port group may be active, and used for programming the FPGA, or they can be reconfigured to act as general purpose I/O. Recovering the configuration port pins for use as general purpose I/O requires you to adhere to the following guidelines:

- You must DISABLE the unused port. You can accomplish this by using the Lattice Radiant Device Constraint Editor under Global tab.
- You must prevent external logic from interfering with device programming.

Table 4.4 lists the shared sysCONFIG pins of the device, and the default state of these pins in user mode. After programming the Nexus device, the default state of the SSPI sysCONFIG pins become general purpose I/O. This means you lose the ability to program the Nexus device using SSPI or Slave I<sup>2</sup>C when using the default sysCONFIG port settings. To retain the SSPI or Slave I<sup>2</sup>C sysCONFIG pins in user mode, be sure to ENABLE them using the Lattice Radiant Device Constraint editor.

Unless specified otherwise, the sysCONFIG pins are powered by the VCCIO0 and VCCIO1 voltage. It is crucial for this to be taken into consideration when provisioning other logic attached to Bank 0 and Bank 1.

The function of each sysCONFIG pin is described in detail.

**Table 4.4. Default State of the sysCONFIG Pins**

| sysCONFIG Pins        |           | Pull During Configuration | Configuration Modes |            |         |         |
|-----------------------|-----------|---------------------------|---------------------|------------|---------|---------|
| Name                  | Type      |                           | JTAG                | MSPI       | SSPI    | I2C/I3C |
| JTAG_ENABLE           | Dedicated | DOWN                      | 1'b1                | 1'bx       | 1'b0    | 1'b0    |
| PROGRAMN              | Shared    | UP                        | 1'b0                | 1'b1       | 1'b0    | 1'b0    |
| INITN                 | Shared    | UP                        | INITN               |            |         |         |
| DONE                  | Shared    | UP                        | DONE                |            |         |         |
| MCLK <sup>3</sup>     | Shared    | UP/DOWN <sup>4</sup>      | —                   | MCLK       | —       | —       |
| MCSN <sup>2</sup>     | Shared    | UP                        | —                   | MCSN       | —       | —       |
| MOSI/MD0              | Shared    | UP                        | —                   | MOSI/D0    | —       | —       |
| MISO/MD1              | Shared    | UP                        | —                   | MISO/D1    | —       | —       |
| MD2                   | Shared    | UP                        | —                   | D2         | —       | —       |
| MD3                   | Shared    | UP                        | —                   | D3         | —       | —       |
| MCSNO/MSDO            | Shared    | UP                        | —                   | MCSNO/MSDO | —       | —       |
| TCK/SCLK              | Shared    | UP                        | TCK                 | —          | SCLK    | —       |
| TMS/SCSN <sup>1</sup> | Shared    | UP                        | TMS                 | —          | SCSN    | —       |
| TDI/SI/SD0            | Shared    | UP                        | TDI                 | —          | MOSI/D0 | —       |
| TDO/SO/SD1            | Shared    | UP                        | TDO                 | —          | MISO/D1 | —       |
| SD2/SCL               | Shared    | UP                        | —                   | —          | D2      | SCL     |
| SD3/SDA               | Shared    | UP                        | —                   | —          | D3      | SDA     |

**Notes:**

1. SCSN should have 4.7 kΩ pull-up resistor on-board for SSPI.
2. MCSN should have 4.7 kΩ pull-up on-board resistor for MSPI.
3. Suggest 1 KΩ pull down resistor on-board for MCLK.
4. Inter weak pull up or pull down is determined by the CR1 Bit 22 (CPOL) setting. UP if CPOL = 1; DOWN if CPOL = 0.

#### 4.4.1. PROGRAMN

PROGRAMN is an input used to configure the FPGA. The PROGRAMN pin is low level sensitive, and has an internal weak pull-up. When PROGRAMN is asserted low, the FPGA exits user mode and starts a device configuration sequence at the Initialization phase, as described earlier. Holding the PROGRAMN pin low prevents the Nexus device from leaving the Initialization phase. The PROGRAMN has a minimum pulse width assertion period ( $t_{\text{PROGRAMN}}$ ) in order for it to be recognized by the FPGA. You can find this minimum time in the AC timing section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).

Be aware of the following special cases when the PROGRAMN pin is active:

- If the device is currently being programmed via JTAG, PROGRAMN is ignored until the JTAG mode programming sequence is complete.
- Toggling the PROGRAMN pin during device configuration interrupts the process and restart the configuration cycle.
- PROGRAMN must not be asserted low until after all power rails have reached stable operation. This can be achieved by either placing an external pull-up resistor and tying it to the VCCIO0 voltage, or permitting the FPGA's internal pull-up resistor to pull the input high.
- PROGRAMN must not make a falling edge transition during the time the FPGA is in the Initialization state to avoid interrupting, restarting or failing configuration. PROGRAMN must be asserted for a minimum low period of  $t_{\text{PROGRAMN}}$  in order for it to be recognized by the FPGA. Failure to meet this requirement can cause the device to become non-operational, requiring power to be cycled.
- For slave port configuration or programming, if the PROGRAMN pin is persisted in user function mode, it has to be driven high before the ISC\_DISABLE command is issued.

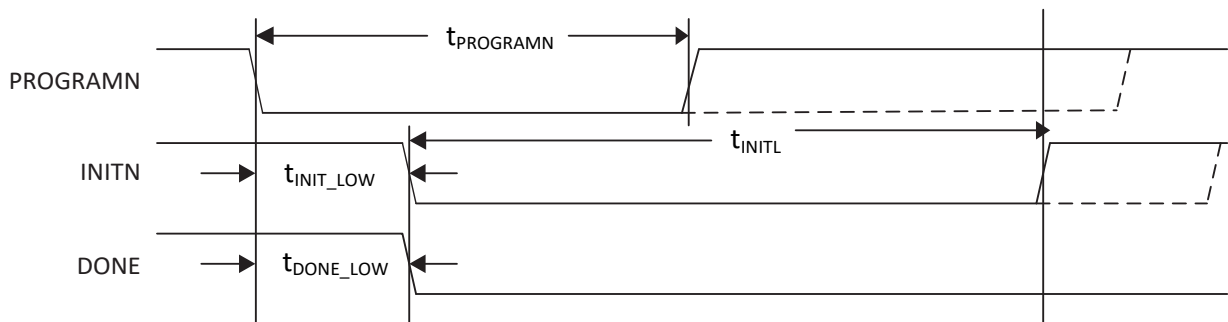


Figure 4.2. Configuration from PROGRAMN Timing

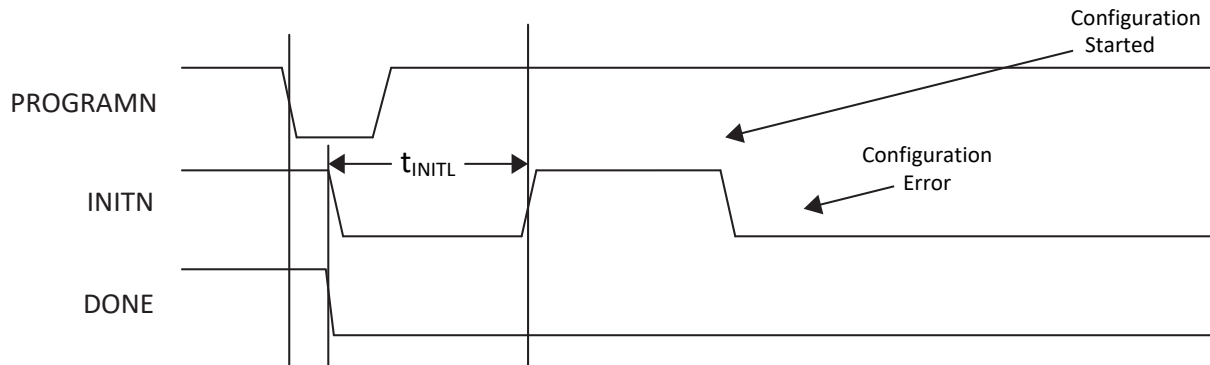
#### 4.4.2. INITN

The INITN pin is a bidirectional open-drain control pin. It has the following functions:

- After power is applied, after a PROGRAMN assertion, or a REFRESH command it goes low to indicate the SRAM configuration memory is being erased. The low time assertion is specified with the  $t_{\text{INIT\_LOW}}$  parameter.
- After the  $t_{\text{INITL}}$  time period has elapsed the INITN pin is deasserted (i.e. is active high) to indicate the Nexus device is ready for its configuration bits. The Nexus device begins loading configuration data from an external SPI Flash.
- INITN can be asserted low by an external agent before the  $t_{\text{INITL}}$  time period has elapsed in order to prevent the FPGA from reading configuration bits. This is useful when there are multiple programmable devices chained together. The programmable device with the longest  $t_{\text{INITL}}$  time can hold all other devices in the chain from starting to get data until it is ready itself.
- The last function provided by INITN is to signal an error during the time configuration data is being read. Once  $t_{\text{INITL}}$  has elapsed and the INITN pin has gone high, any subsequent INITN assertion signals the Nexus device has detected an error during configuration.

The following conditions cause INITN to become active, indicating the Initialization state is active:

- Power has just been applied
- PROGRAMN falling edge occurred
- The IEEE 1532 REFRESH command is sent using a slave configuration port (JTAG or SSPI). If the INITN pin is asserted due to an error condition, the error can be cleared by correcting the configuration bitstream and forcing the FPGA into the Initialization state.



**Figure 4.3. Configuration Error Notification**

If an error is detected when reading the bitstream, INITN goes low, the internal DONE bit is not set, the DONE pin stays low, and the device does not wake up. The device configuration fails when the following happens:

- The bitstream CRC error is detected
- The invalid command error is detected
- A time out error is encountered when loading from the external Flash. This can occur when the device is in MSPI configuration mode and the SPI Flash device is not programmed.
- The program done command is not received when the end of on-chip SRAM configuration or external Flash memory is reached

#### 4.4.3. DONE

The DONE pin is a bi-directional open drain with a weak pull-up that signals the FPGA is in user mode. DONE is first able to indicate entry into user mode only after an internal DONE bit is asserted. The internal DONE bit defines the beginning of the FPGA Wake-Up state.

The FPGA can be held from entering User mode indefinitely by having an external agent keep the DONE pin asserted low. A common reason for keeping DONE driven low is to allow multiple FPGAs to be completely configured. As each FPGA reaches the DONE state, it is ready to begin operation. The last FPGA to configure can cause all FPGAs to start in unison.

The DONE pin drives low in tandem with the INITN pin when the FPGA enters Initialization mode. As described earlier, this condition happens when power is applied, PROGRAMN is asserted, or an IEEE 1532 Refresh command is received via an active configuration port.

Sampling the DONE pin is a way for an external device to tell if the FPGA has finished configuration.



## 4.5. Master SPI sysCONFIG Pins

The following is a list of the dual-purpose Master SPI sysCONFIG pins. If any of these pins are used for configuration and for user I/O, you must adhere to the requirements listed at the start of the Configuration pin sections. These pins are powered by VCCIO0.

### 4.5.1. MCLK

The MCLK, when active, is clock used to sequentially load the configuration data for the FPGA. When Master Configuration mode is used, MCLK becomes an output clock with the frequency that you set. The output is used to drive to external memory device. The maximum MCLK frequency and the data setup/hold parameters can be found in the AC timing section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#). MCLK actively drives until all the configuration data is received. When the Nexus device enter user mode the MCLK output tristates. The MCLK is reserved for use in MSPI mode, in most post-configuration applications, as the reference clock for performing memory transactions with the external SPI PROM. See [Master SPI Modes](#) section for details.

The Nexus device generates MCLK from an internal oscillator. The initial frequency of the MCLK is nominally 3.5 MHz. The MCLK frequency can be altered using the MCCLK\_FREQ parameter. You can select the MCCLK\_FREQ using the Lattice Radiant Device Constraint Editor. For a complete list of the supported MCLK frequencies, see [Table 4.5](#).

**Table 4.5. Nexus MCLK Valid Frequencies**

| MCLK Frequency (MHz) |
|----------------------|
| 3.5                  |
| 7.0                  |
| 14.1                 |
| 28.1                 |
| 56.2                 |
| 90                   |
| 112.5                |
| 150                  |

At startup, the lowest frequency MCLK is used by the FPGA. During the initial stages of device configuration, the frequency value specified using MCCLK\_FREQ contained in the bitstream is loaded into the FPGA. Once the Nexus device accepts the new MCLK\_FREQ value, the MCLK output begins driving the selected frequency. Make certain when selecting the MCLK\_FREQ that you do not exceed the frequency specification of your configuration memory, or of your PCB. When making MCLK\_FREQ decisions, review the AC specifications in [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).

### 4.5.2. MCSN

MCSN\_ – In MSPI mode, the MCSN becomes a low true Chip Select output that drives the SPI Serial Flash chip select. If SPI memory needs to be accessed using the SPI port while the part is in user mode (the DONE pin is high) then the MASTER\_SPI\_PORT= ENABLE, must be used to preserve this pin as MCSN. When the Nexus device is not in MSPI mode, the MCSN is a general purpose I/O with a weak pulldown. Adding a 4.7 kΩ to 10 kΩ pull-up resistor to MCSN pin on the Nexus device is recommended. MCSN must ramp in tandem with the SPI PROM VCC input. It remains a general purpose I/O when the FPGA enters user mode.

### 4.5.3. MOSI/MD0

MOSI/MD0 – In Master SPI configuration mode, the MOSI pin is the serial data output for SPI command and data. It becomes D0 of the data bus in Dual or Quad mode.

#### 4.5.4. MISO/MD1

MISO/MD1 – In Master SPI configuration mode, the MISO pin is the serial data input. It becomes D1 of the data bus in Dual or Quad mode.

#### 4.5.5. MD2

MD2 – In Master SPI configuration mode, this bit becomes the D2 of the data bus in Quad mode.

#### 4.5.6. MD3

MD3 – In Master SPI configuration mode, this bit becomes the D3 of the data bus in Quad mode.

#### 4.5.7. MCSNO/MSDO

This is an output pin for configuration daisy chain support, which has the following purposes:

MCSNO – For configuration daisy chaining implemented with the Flow-through attribute, this attribute allows the MCSNO pin to be driven when the done bit is set and configuration of the first device is complete. The MCSNO of the first device drives the CSN of the second part.

MSDO – For configuration daisy chaining implemented with the Bypass mode, this pin becomes a data output pin for serial daisy chaining. When the device is fully configured, the Bypass instruction contained within the bitstream is executed and the data on MOSI is then routed to the DOUT pin. This allows data to be passed, serially, to the next device. While using Dual or Quad mode for the upstream device, the rate of the data sent to upstream device needs to slow down to allow the data to be serially shifted to downstream device.

### 4.6. Slave sysCONFIG Pins

#### 4.6.1. TCK/SCLK

This pin is used by both JTAG and Slave SPI configuration interface.

TCK – If the JTAG port is enabled, this pin serves as the clock pin for the JTAG interface. The test clock pin (TCK) provides the clock used to time the other JTAG port pins. Data is shifted into the instruction or data registers on the rising edge of TCK and shifted out on the falling edge of TCK. The TAP is a static design permitting TCK to be stopped in either the high or low state. The maximum input frequency for TCK is specified in the DC and Switching Characteristics section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#). The TCK pin does not have a pull-up. An external pull-down resistor of 4.7 kΩ is recommended to avoid inadvertently clocking the TAP controller as power is applied to the Nexus device.

SCLK – In slave SPI mode, this pin is the clock input for the slave SPI configuration interface. The maximum CCLK frequency and the data setup/hold parameters can be found in the AC timing section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).

#### 4.6.2. TMS/SCSN

This pin is used by both JTAG and Slave SPI configuration interface.

TMS – If the JTAG port is enabled, this pin serves as the Test Mode Select pin for the JTAG interface. The Test Mode Select (TMS) pin is an input pin that controls the progression through the 1149.1 compliant state machine states. The TMS pin is sampled on the rising edge of TCK. The JTAG state machine remains in or transitions to a new TAP state depending on the current state of the TAP, and the present state of the TMS input. An internal pull-up resistor is present on TMS per the JTAG specification.

SCSN – In slave SPI mode, this pin is the active low chip select input for the slave SPI configuration interface. A 4.7 kΩ external pull-up resistor is recommended.

### 4.6.3. TDI/SI/SD0

This pin is used by both JTAG and Slave SPI configuration interface.

TDI – If the JTAG port is enabled, this pin serves as the Test Data Input (TDI) pin, which is used to shift in serial test instructions and data. This pin should be wired to TDI of the JTAG connector, or to TDO of an upstream device in a JTAG chain. An internal pull-up resistor on the TDI pin is provided.

SI/SD0 – In Slave SPI configuration mode, the SI pin is the serial data input for SPI command and data. It becomes D0 of the data bus in Dual or Quad mode.

### 4.6.4. TDO/SO/SD1

This pin is used by both JTAG and Slave SPI configuration Interface.

TDO – If the JTAG port is enabled, this pin serves as the Test Data Output (TDO) pin, which is used to shift out serial test instructions and data. When TDO is not being driven by the internal circuitry, the pin is in a high impedance state. The only time TDO is not in a high impedance state is when the JTAG state machine is in the Shift IR or Shift DR state. This pin should be wired to TDO of the JTAG connector, or to TDI of a downstream device in a JTAG chain. An internal pull-up resistor on the TDO pin is provided.

SO/SD1 – In Slave SPI configuration mode, the SO pin is the serial data output for SPI data. It becomes D1 of the data bus in Dual or Quad mode.

### 4.6.5. SD2/SCL

This pin is used by both Slave SPI and Slave I<sup>2</sup>C/I<sup>3</sup>C configuration Interface.

SD2 – In Slave SPI configuration mode, this bit becomes the D2 of the data bus in Quad mode.

SCL – In Slave I<sup>2</sup>C/I<sup>3</sup>C configuration mode, this is the serial clock line of the I<sup>2</sup>C or I<sup>3</sup>C bus.

### 4.6.6. SD3/SDA

This pin is used by both Slave SPI and Slave I<sup>2</sup>C/I<sup>3</sup>C configuration Interface.

SD3 – In Slave SPI configuration mode, this bit becomes the D3 of the data bus in Quad mode.

SDA – In Slave I<sup>2</sup>C/I<sup>3</sup>C configuration mode, this is the serial data line of the I<sup>2</sup>C or I<sup>3</sup>C bus.

## 4.7. PERSISTENT

The internal PERSISTENT control bits are used to determine whether the dual-purpose Master and Slave sysCONFIG pins remain as sysCONFIG pins during normal operation, for example, to support transparent programming or configuration. The Nexus device has several PERSISTENT physical SRAM cells that determine the existence of the Master SPI port, Slave SPI port or I<sup>2</sup>C/I<sup>3</sup>C port after entering user mode. These settings can be set in Lattice Radiant Device Constraint Editor under Global tab as shown in [Table 4.6](#).

**Table 4.6. sysCONFIG Pins Global Preferences**

| Port Setting      | Value  | Pins Affected                            | Details  |
|-------------------|--------|--|--|
| SLAVE_SPI_PORT    | SERIAL | MCLK, MCSN, MOSI/MD0, MISO/MD1           | If enabled, persisted for configuration purpose. |
|                   | DUAL   | MCLK, MCSN, MOSI/MD0, MISO/MD1           |  |
|                   | QUAD   | MCLK, MCSN, MOSI/MD0, MISO/MD1, MD2, MD3 |  |
| MASTER_SPI_PORT   | SERIAL | SCLK, SCSN, SI/SD0, SO/SD1               | If enabled, persisted for configuration purpose. |
|                   | DUAL   | SCLK, SCSN, SI/SD0, SO/SD1               |  |
|                   | QUAD   | SCLK, SCSN, SI/SD0, SO/SD1, SD2, SD3     |  |
| SLAVE_I2CI3C_PORT | ENABLE | SCL, SDA                                 | If enabled, persisted for configuration purpose. |

**Note:** JTAG Persist follows JTAG enable pin.

## 5. Master Port Configuration Process and Flow

Prior to becoming operational, the FPGA goes through a sequence of states, including initialization, configuration and wake-up.

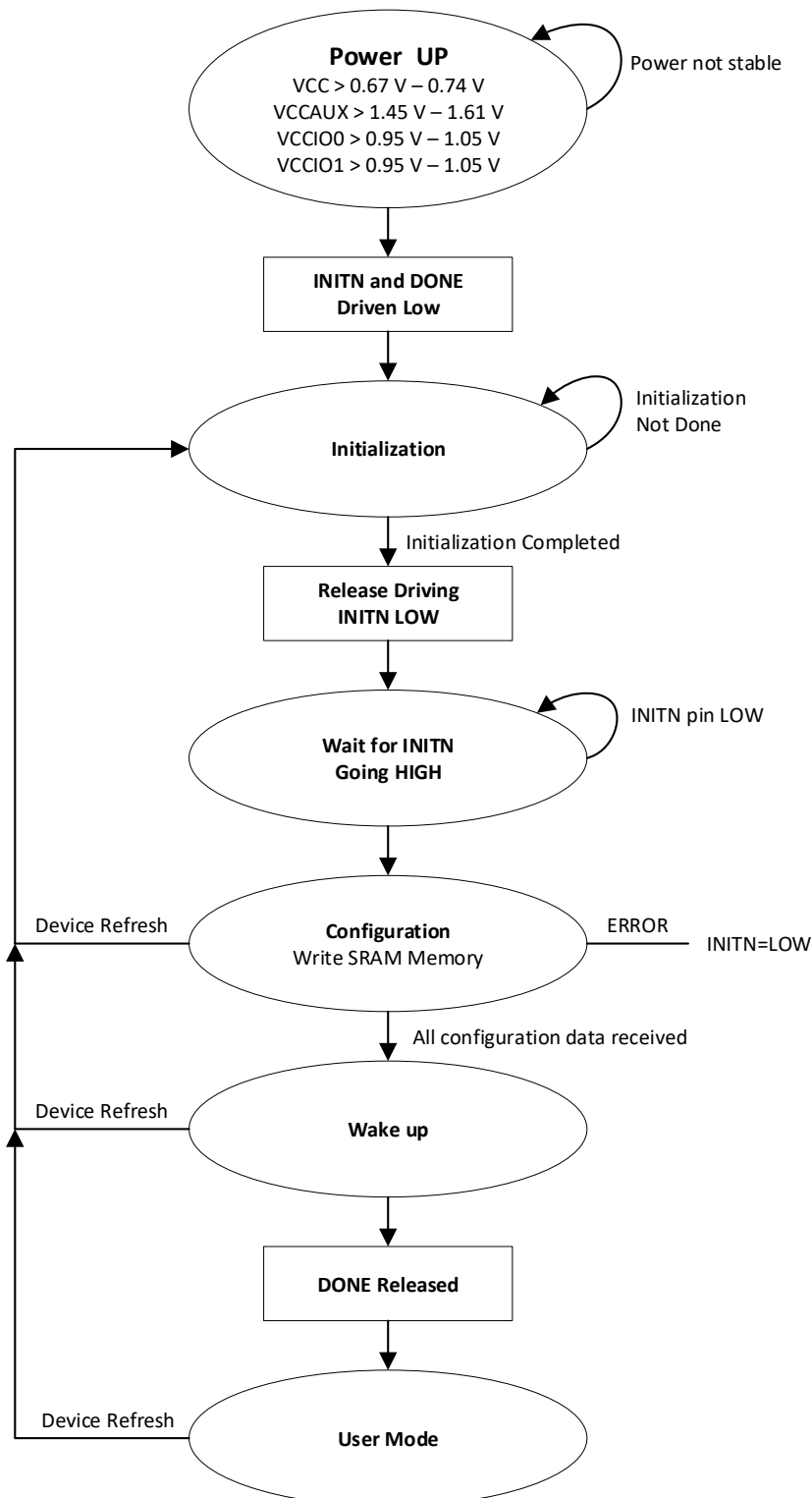


Figure 5.1. Master Port Configuration Flow

The Nexus device sysCONFIG ports provide industry standard communication protocols for programming and configuring the FPGA. Each of the protocols shown in Table 4.2 provides a way to access the configuration SRAM of the Nexus device. The Configuration Ports Arbitration section provides information about the availability of each sysCONFIG port.

## 5.1. Power-up Sequence

In order for the Nexus device to operate, power must be applied to the device. During a short period of time, as the voltages applied to the system rise, the FPGA stays in an indeterminate state.

As power continues to ramp, a Power-On-Reset (POR) circuit inside the FPGA becomes active. The POR circuit, once active, makes sure the external I/O pins are in a high-impedance state. It also monitors the VCC, VCCAUX, VCCIO0, and VCCIO1 input rails. The POR circuit waits for the following conditions:

- $VCC > 0.67\text{ V} - 0.74\text{ V}$
- $VCCAUX > 1.45\text{ V} - 1.61\text{ V}$
- $VCCIO0 > 0.95\text{ V} - 1.05\text{ V}$
- $VCCIO1 > 0.95\text{ V} - 1.05\text{ V}$

When these conditions are met the POR circuit releases an internal reset strobe, allowing the device to begin its initialization process. The Nexus device asserts INITN active low, and drives DONE low. When INITN and DONE are asserted low, the device moves to the initialization state, as shown in Figure 5.1.

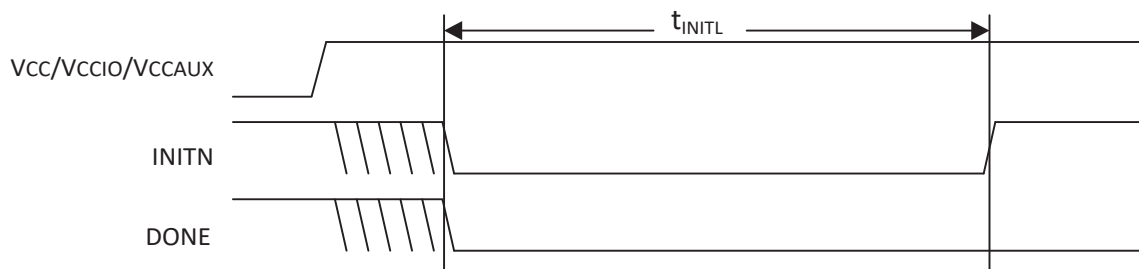


Figure 5.2. Configuration from Power-On-Reset Timing

## 5.2. Initialization

The Nexus device enters the memory initialization phase immediately after the Power On Reset circuit drives the INITN and DONE status pins low. The purpose of the initialization state is to clear all of the SRAM memory inside the FPGA.

The FPGA remains in the initialization state until all of the following conditions are met:

- The  $t_{INITL}$  time period has elapsed
- The PROGRAMN pin is deasserted
- The INITN pin is no longer asserted low by an external master

The dedicated INITN pin provides two functions during the initialization phase. The first is to indicate the FPGA is currently clearing its configuration SRAM. The second is to act as an input preventing the transition from the initialization state to the configuration state.

During the  $t_{INITL}$  time period the FPGA is clearing the configuration SRAM. When the Nexus device is part of a chain of devices each device has different  $t_{INITL}$  initialization times. The FPGA with the slowest  $t_{INITL}$  parameter can prevent other devices in the chain from starting to configure. Premature release of the INITN in a multi-device chain may cause configuration of one or more chained devices to fail to configure intermittently.

The active-low, open-drain initialization signal INITN must be pulled high by an external resistor when initialization is complete. To synchronize the configuration of multiple FPGAs, one or more INITN pins should be wire-ANDed. If one or more FPGAs or an external device holds INITN low, the FPGA remains in the initialization state.

The GPIO of the device at power-up defaults to tri-stated outputs with active weak input pull-downs. After configuration, all GPIO included in the user design wake up in the user-defined condition. GPIO not defined in the user design remain output tri-stated and the input with a weak pull-down enabled. This default avoids inadvertent effects of the inputs rising while powering up. In some cases, this can cause a problem if other connected devices on the board reset or trigger from an active high signal.

Before/during configuration, the dedicated JTAG\_ENABLE pin has pull-down, and dual-purpose sysCONFIG I/O with pull-up, which are excluded from the GPIO default setting.

### 5.3. Configuration

The releasing HIGH on the INITN pin causes the FPGA to enter the configuration state. The FPGA is able to accept the configuration bitstream created by the Lattice Radiant Software.

The Nexus device begins fetching configuration data from non-volatile memory. The Nexus device does not leave the Configuration state if there is no valid configuration data.

INITN is used to indicate an error exists in the configuration data. When INITN is active, high configuration is proceeding without issue. If INITN is asserted low, an error has occurred and the FPGA does not operate.

### 5.4. Wake-up

Wake-up is the transition from configuration mode to user mode. The Nexus device's fixed four-phase wake-up sequence starts when the device has correctly received all of its configuration data. You can arrange the order of these four phases to meet specific implementation requirements. When all configuration data is received, the FPGA asserts an internal DONE status bit. The assertion of the internal DONE causes a Wake Up state machine to run that sequences four controls. The four control strobes are:

- Global Set/Reset (GSR)
- Global Output Enable (GOE)
- External DONE
- Global Write Disable (GWDISn)

One phase of the Wake-Up process is for the FPGA to release the Global Output Enable. When it is asserted, permits the FPGA's I/O to exit a high-impedance state and take on their programmed output function. The FPGA inputs are always active. The input signals are prevented from performing any action on the FPGA flip-flops by the assertion of the Global Set/Reset (GSR).

Other phases of the Wake-Up process release the Global Set/Reset and the Global Write Disable controls.

The Global Set/Reset is an internal strobe that, when asserted, causes all I/O flip-flops, Look Up Table (LUT) flip-flops, distributed RAM output flip-flops, and Embedded Block RAM output flip-flops that have the *GSR enabled* attribute to be set/cleared per their hardware description language definition.

The Global Write Disable is a control that overrides the write enable strobe for all RAM logic inside the FPGA. As mentioned previously, the inputs on the FPGA are always active. Keeping GWDIS asserted prevents accidental corruption of the instantiated RAM resources inside the FPGA.

Another phase of the Wake-Up process asserts the external DONE pin. The external DONE is a bi-directional, open-drain I/O only when it is enabled. An external agent that holds the external DONE pin low prevents the wake-up process of the FPGA from proceeding. Only after the external DONE, if enabled, is active high does the final wake-up phase complete. Wake-Up completes uninterrupted when the external DONE pin is not enabled.

Once the final wake-up phase is complete, the FPGA enters user mode. This process is detailed later in the document.

## 5.5. Early I/O Release

The Nexus device supports an Early I/O Release feature, which allows the I/O that reside in the I/O Banks on the left and right of the device (I/O Bank1, I/O Bank2, I/O Bank6, and I/O Bank 7), to assume user-defined drive states at the beginning of bistream processing. The Early I/O Release feature releases the I/O after processing the I/O configuration for the left and right banks, which is located near the head of the bitstream data. Once data is programmed in either bank the I/O is released to a predefined state. This feature is enabled by setting the EARLY\_IO\_RELEASE preferences to ON in the Lattice Radiant Device Constraint Editor.

In addition, Early I/O Release requires you to instantiate an output buffer register with an asynchronous set or reset function, to indicate the desired drive 1 or drive 0 behavior, respectively, during the Early Release period. Unregistered outputs in Early-Release banks drive High-Z until full device configuration is complete. Be aware that some of the I/O in Bank 1, including the dual-purpose sysCONFIG I/O, cannot be utilized as Early Released I/O. Also, if the ECDSA bitstream authentication is enable for the Nexus device, the Early I/O Release feature is not supported.

## 5.6. User Mode

The Nexus device enters user mode immediately following the Wake-Up sequence has completed. User Mode is the point in time when the Nexus device begins performing the logic operations you designed. The device remains in this state until one of three events occurs:

- The PROGRAMN input pin is asserted
- A REFRESH command is received via one of the configuration ports
- Power is cycled or power supply levels drop below their specified trigger levels

## 5.7. Clearing the Configuration Memory and Re-initialization

The current user mode configuration of the Nexus device remains in operation until they are actively cleared, or power is lost. Several methods are available to clear the internal configuration memory of the Nexus device. The first is to remove power and reapply power. Another method is to toggle the PROGRAMN pin. Lastly, you can reinitialize the memory through a Refresh command. Any active configuration port can be used to send a Refresh command.

Invoking one of these methods causes the Nexus device to drive INITN and DONE low. The Nexus device enters the initialization state as described earlier.

## 5.8. Reconfiguration Priority

There are many sources that can initiate a reconfiguration while a configuration is already in process. When initiating a reconfiguration, the sources are prioritized depending on which of them initiated the original configuration. Note that if an interruption occurs, the reconfiguration occurs without informing the original configuration source that the configuration did not complete. JTAG always has the highest priority and any JTAG initiated configuration event causes a reconfiguration to occur. Toggling the PROGRAMN pin has the next highest priority. It interrupts any current configuration other than a JTAG configuration.

## 6. Configuration Modes

The Nexus device provides multiple options for loading the configuration SRAM from a non-volatile memory. The previous section describes the physical interface necessary to interact with the configuration logic of the Nexus device. This section focuses on describing the functionality of each of the different configuration modes. Descriptions of important settings required in the Lattice Radiant Device Constraint Editor View are also discussed.

### 6.1. Master SPI Modes

The Master SPI port is considered an intelligent port and it is the default mode before any other slave configuration port is activated. It is capable of performing read and write actions based on the command shifted into the device. All commands are built inside the bitstreams. In Master SPI mode (MSPI), the Nexus device begins retrieving configuration data from the SPI Flash when power is applied, a REFRESH command is received, or the PROGRAMN pin is asserted and released. One MSPI mode CFGMDN setting supports several different submodes such as SLOW or FAST SPI speed, DUAL-boot and MULTI-boot, as well as supporting DUAL and QUAD read SPI Flash devices. All the different submodes are supported by different commands build inside the bitstream. Lattice Radiant flow only generates default submode bitstream, which is serial and slow read. To change to a different submode, you must alter the bitstream using Lattice Deployment Tool. The MCLK/CCLK I/O takes on the Master Clock (MCLK) function, and begins driving a nominal 3.5 MHz clock to the SPI Flash's SCLK input. MCSN is asserted low, commands are transmitted to the PROM over the MOSI/MD0 output, and data is read from the PROM on the MISO/MD1 input pin. When all of the configuration data is retrieved from the PROM, the MCSN pin is deasserted, and the MSPI output pins are tristated.

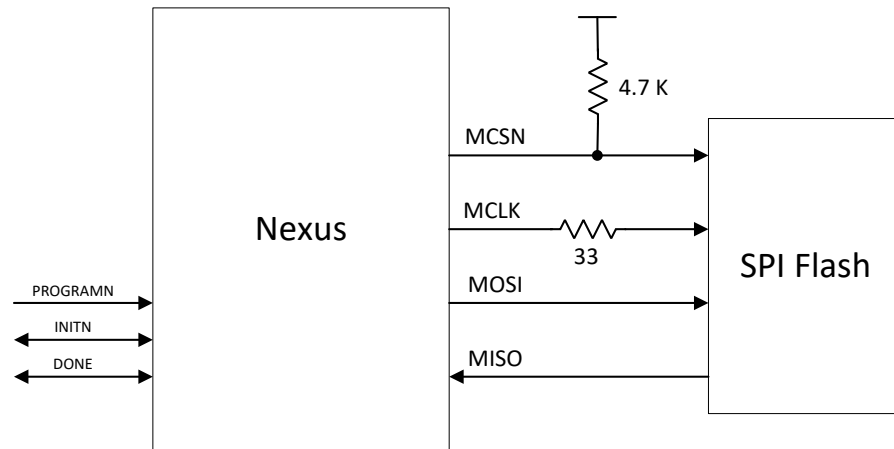
**Table 6.1. Master SPI Configuration Port Pins**

| Pin Name | Function | Direction                | Description  |
|----------|----------|--------------------------|--|
| MCLK     | MCLK     | Output with weak pull-up | Master clock used to time data transmission/reception from the Nexus device Configuration Logic to a slave SPI PROM. |
| MCSN*    | MCSN     | Output                   | Chip select used to enable an external SPI PROM containing configuration data.                                       |
| MOSI/MD0 | MOSI     | Output                   | Carries output data from the Nexus device Configuration Logic to the slave SPI PROM.                                 |
|          | D0       | Input                    | Input pin for bitstream data as DUAL and QUAD read mode  |
| MISO/MD1 | MISO     | Input                    | Carries input data from the slave SPI PROM to the Nexus device Configuration Logic.                                  |
|          | D1       | Input                    | Input pin for bitstream data as DUAL and QUAD read mode.   |
| MD[3:2]  | D[3:2]   | Input                    | This is the input pins for bitstream data from SPI Flash, used only on QUAD read mode.                               |

\*Note: Use 4.7 kΩ pull-up resistor.

The MCLK frequency always starts downloading the configuration data at the nominal 3.5 MHz frequency. The MCCLK\_FREQ parameter, accessed using Device Constraint Editor, can be used to increase the configuration frequency. The configuration data in the PROM has some padding bits, and then the data altering the MCLK base frequency is read. The Nexus device reads the remaining configuration data bytes using the new MCLK frequency. For higher MCLK frequency with fast slew rate, a 33 Ω damping resistor is recommended.





**Figure 6.1. Nexus Master SPI Port with SPI Flash**

Once the SPI Flash contains your configuration data, you can test the configuration. Assert the PROGRAMN, transmit a REFRESH command, or cycle power to the board, and the Nexus device configures from the external SPI Flash.

### 6.1.1. Method to Enable the Master SPI Port

The Master SPI port is enabled by default if there is no slave configuration port enabled.

- When the device is powered up, or when the PROGRAMN pin is toggled, or when the REFRESH command is executed, the Master SPI port is selected as the configuration port by default. A port is said to be a configuration port when it is capable of executing both bitstream write and read commands. This is the only method with which you can perform DUAL read and QUAD read from SPI Flash.
- Enabling Master SPI port persistence.

The MSPI port could be persisted in user mode by setting the desired Value (SERAL, DUAL or QUAD) for MASTER\_SPI\_PORT in Lattice Radiant Device Constraint Editor. Once set, the configuration bitstream contains optional Master SPI persistence bits. When the device completes configuration and wakes up, it checks the persistent bits to determine if the Master SPI port is to remain operational once in user mode. This selection is independent of the CFG port arbitration during configuration phase. A port enabled by persistence is a Transparent Mode port. It reserves those pins from being occupied by user logic.

Note that both the DONE pin and the INITN pin must be high. If not, then the device is not in user mode. The persistent bits have no affect when the device is not in user mode.

### 6.1.2. Dual-Boot and Multi-Boot Configuration Modes

Both the primary and the golden (fail-safe) configuration data is stored in external SPI memory. The fail-safe pattern is available in case the primary pattern would fail to load. The primary image can fail in one of the following reasons:

- A time-out error is encountered while waiting for PREAMBLE code
- Device ID checking failed at the beginning of the bitstream.
- An illegal command is encountered.
- A bitstream CRC error is detected
- A time-out error is encountered when loading from the primary pattern stored in the external memory

A CRC error is caused by incorrect data being written into the SRAM configuration memory. Data is read from the external Flash memory. As data enters the Configuration Engine the data is checked for CRC consistency. Before the data enters the Configuration SRAM the CRC must be correct. Any incorrect CRC causes the device to erase the Configuration SRAM and retrieve configuration data from the external golden pattern.

It is possible for the data to be correct from a CRC calculation perspective, but not be functionally correct. In this instance the internal DONE bit never becomes active. The Nexus device counts the number of master clock pulses it provided after the Power On Reset signal is released. When the count expires without DONE becoming active, the FPGA attempts to get its configuration data from the external golden pattern.

Dual boot configuration mode typically requires two configuration data files. One of the two configuration data files is a fail-safe image that is rarely, if ever, updated. The second configuration data file is a working image that is routinely updated. Both the working (primary) image and the fail-safe image are stored in the external SPI memory. One Lattice Radiant project can be used to create both the working and the fail-safe configuration data files. Configure the Lattice Radiant project with an implementation named *working*, and an implementation named *failsafe*. Read the Lattice Radiant Online Help for more information about using Lattice Radiant implementations.

The Nexus device supports dual-boot with the MSPI mode. If the primary pattern fails to load correctly, the Nexus device starts loading data from the golden sector in the SPI Flash device. A blank external Flash device causes a dual-boot event failure indicated by INITN going low. This is due to the absence of a primary or golden boot image.

The dual boot feature allows you to split a SPI Flash device into two sections, the first containing a sacred *golden boot* file, and a second updatable *primary boot* file, which can be erased and reprogrammed. By default, the FPGA loads the primary boot file in block 0 (0x000000). If the FPGA fails in configuration, it automatically loads the golden boot file in the last block (0xFFFF00). This allows your system to boot to a known operable state, so that it continues to operate if for some reason (such as a power failure) the SPI Flash fails to program correctly.

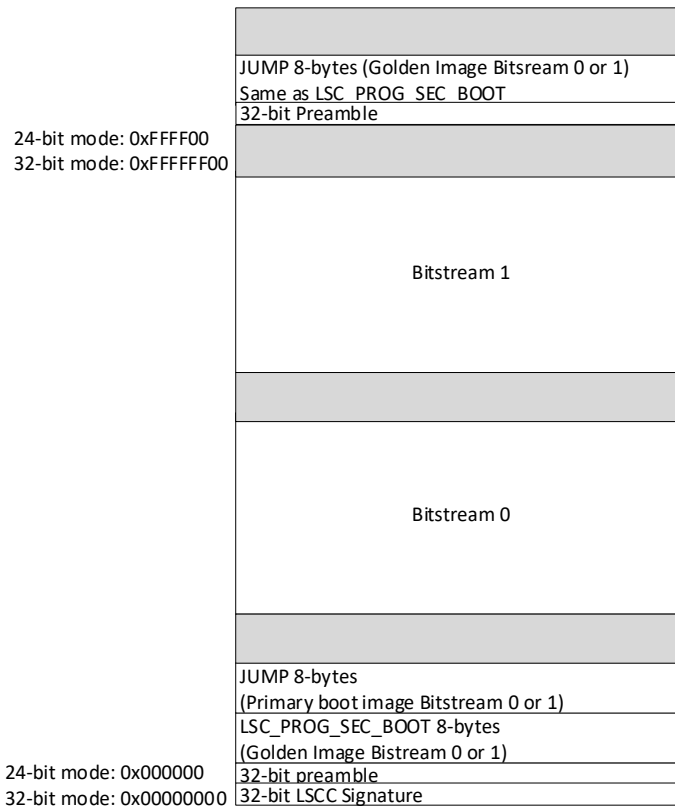
You can dynamically switch between up to five different design revisions stored in external Flash. Multi-boot, or the ability to dynamically reconfigure from multiple design bitstreams, is similar to the dual-boot where there is one primary (working) bitstream and up to four alternate bitstreams.

For multi-boot operation, the next target address is set in memory that was loaded during the current configuration memory load. Initiating reprogramming by toggling the PROGRAMN pin or issuing a REFRESH via any sysCONFIG port causes the device to load from the defined SPI Flash address. Dual-boot can also be deployed with multi-boot, allowing a golden (fail-safe) design (or sixth design) to be available in the external Flash. For detail information regarding multi-boot operation, refer to [Multi-Boot Usage Guide for Nexus Platform \(FPGA-TN-02145\)](#).

Lattice Radiant Deployment Tool allows you to assemble SPI Flash images formatted to correctly match the hardware sector mapping.

### 6.1.3. Ping-Pong Boot

The Ping-Pong boot mode utilize the JUMP table to select an image for booting without changing location of the image in flash. This is done with a jump table starting at address 0x000000 in flash containing two instructions LSC\_PROG\_SEC\_BOOT and JUMP. For backup in case, the jump table were to become corrupted by a power failure. A backup JUMP instruction should be programmed at offset 0xFFFF00 pointing to the golden boot image. Refer to [Figure 6.2](#) for the two images stored in flash Bitstream 0 and Bitstream 1. The secondary bitstream offset is programmed both with the command LSC\_PROG\_SEC\_BOOT located in the jump table starting at offset 0x000000 and the JUMP instruction located in the backup jump table at offset 0xFFFF00, and the primary bitstream is selected with the JUMP command in the jump table. In order to swap the order of booting these three instructions are all that needs to be re-programmed.



**Figure 6.2. Jump Table**

#### 6.1.4. Dual and Quad Master SPI Read Mode

The master SPI configuration mode in the Nexus device is expanded to support new industry standard Quad I/O SPI Flash memory. The support of (Serial Multi I/O) Flash memory enables fast parallel read.

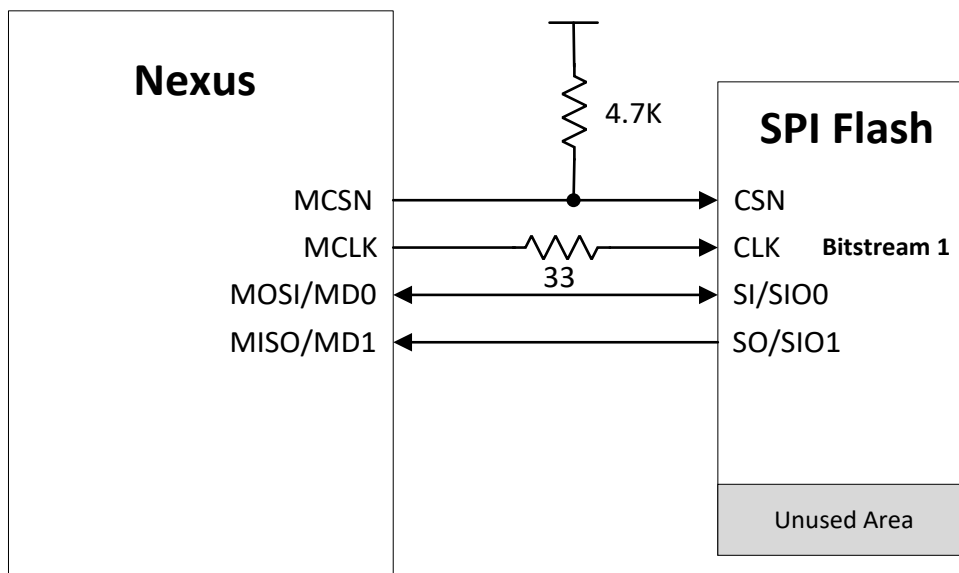
A typical SPI Flash interface uses either 4 or 6 interface signals to the FPGA. The Standard SPI flash uses CLK, CS, SI, and SO while Quad SPI Flash uses CLK, CS, I/O0, I/O1, I/O2 and I/O3, maintaining function and pin-out compatibility with the single SPI Flash devices, while adding Dual-I/O and Quad-I/O SPI capability. All SPI modes are submodes of MASTER SPI, therefore there is no longer a separate CFGMDN pin decode for each SPI mode.

In Dual mode, the Fast-Read Dual Output (BBh) instruction is issued and is similar to the standard Fast Read (0Bh) instruction except that data is output on two pins, SO and SIO0, instead of just SO. This allows data to be transferred from the dual output at twice the rate of standard SPI devices. In QUAD mode, the Fast-Read Quad Output (EBh) instruction is issued and is similar to the standard Fast Read (0Bh) instruction except that data is output on four data pins, instead of just SO. This allows data to be transferred from the quad output at four times the rate of standard SPI devices.

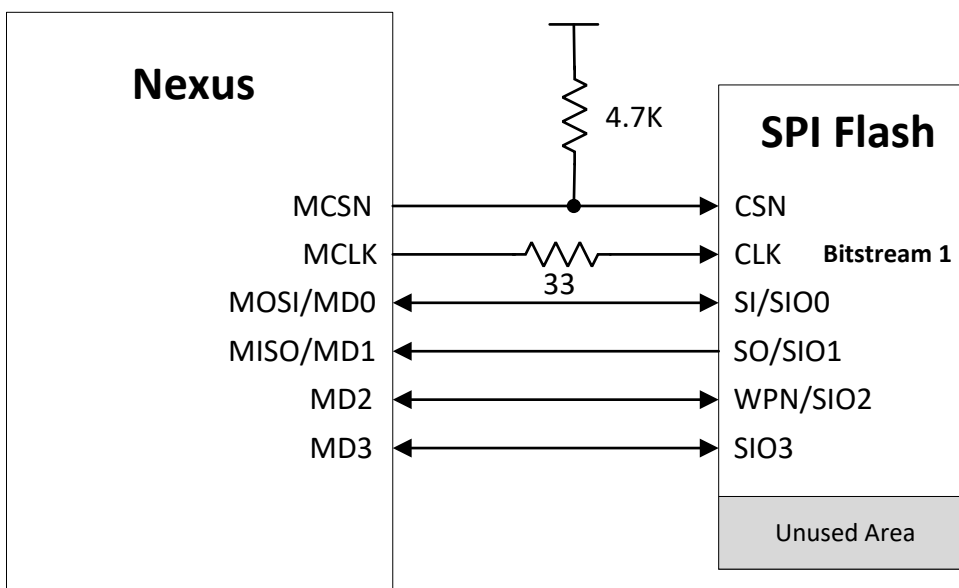
To change the SPI read mode to fast read, dual read or quad read, the Deployment Tool must be used to generate the hex file used for programming the SPI flash device. Lattice Radiant flow only generates bitstreams with default SPI read mode, which is slow serial (03 h), read mode. The sysCONFIG option in Lattice Radiant Device Constraint Editor is for the software to preserve appropriated pins to help customer design flow. It does not serve the purpose of enabling SPI port nor setting appropriate bits in the PROM file.

**Table 6.2. Dual/Quad SPI Port Names**

| Nexus Devices Pin Name | Flash Device Pin | MSPI Function DUAL(D), QUAD(Q)   |
|------------------------|------------------|--|
| MCLK                   | SCK              | Clock output from the Nexus Device Configuration Logic and Master SPI controller. Connect MCLK to the SCLK input of the Slave SPI device. (D)(Q) |
| MD [3:2]               | SIO[3:2]         | Data I/O between the Nexus device and SPI device. (Q)  |
| MD[1:0]                | SIO[1:0]         | Data I/O between the Nexus device and SPI device. (D)(Q)   |
| MCSN                   | CSN              | Chip select output from the Nexus device configuration logic to the slave SPI Flash holding configuration data for the Nexus device. (D)(Q)      |



**Figure 6.3. One Dual SPI Flash Interface (Dual)**



**Figure 6.4. One Quad SPI Flash Interface (Quad)**

## 6.2. Slave SPI Mode

The Nexus device provides a Slave SPI (SSPI) configuration port that allows you to access features provided by the Configuration Logic. You can reprogram the Configuration SRAM and access status/control registers within the Configuration Logic block. The external Flash memory updates are done using either offline or transparent operations. It is necessary to send a REFRESH command to load a new external Flash image into the configuration SRAM. When the Nexus device is in Transparent Mode, only read type commands for the Configuration SRAM are supported, allowing for device verification or debugging. The command set consists of 8-bit opcodes. Some of the commands have a 24-bit operand following the 8-bit opcode. The Slave SPI port is a byte bounded port; all input and output data must be byte bounded.

In the Slave SPI mode, the TCK/SCLK pin becomes SCLK (such as Slave SPI Clock). Input data is read into the Nexus device on the MOSI pin at the rising edge of SCLK. Output data is valid on the MISO pin at the falling edge of SCLK. The SCSN acts as the chip select signal. When SCSN is high, the SSPI interface is deselected and the MISO pin is tristated. Commands can be written into and data read from the Nexus device when SCSN is asserted.

The SSPI port can be activated through the CFG port arbitration procedure before the device enters user mode. The SSPI port can be persisted in user mode by setting the desired Value (SERAL, DUAL or QUAD) for SLAVE\_SPI\_PORT in Lattice Radiant Device Constraint Editor.

Using the SSPI port simplifies the Nexus device configuration process. Lattice provides C source code called sspiembedded to insulate you from the complexity of programming the Nexus device. Refer to the Lattice Radiant online help about sspiembedded.

**Table 6.3. Slave SPI Configuration Port Pins**

| Pin name   | Function | Direction               | Description  |
|------------|----------|-------------------------|--|
| TCK/SCLK   | SCLK     | Input with weak pull-up | Clock used to time data transmission/reception from an external SPI master device to the Nexus device Configuration Logic.   |
| SCSN*      | CSN      | Input with weak pull-up | Nexus device Configuration Logic slave SPI chip select input. SN is an active low input. High to Low transition: reset the device, prepare it to receive commands.<br>Low to High transition: Completes or terminates the current command. |
| TDI/SI/SD0 | MOSI     | Input                   | Carries output data from the external SPI master to the Nexus device Configuration Logic   |
|            | D0       | Inout                   | D0 of the data bus for DUAL and QUAD mode  |
| TDO/SD1    | MISO     | Output                  | Carries output data from the Nexus Device Configuration Logic to the external SPI master. It is normally tristate with an internal pull-up, It becomes active only when the command is a read type command.                                |
|            | D1       | Inout                   | D1 of the data bus for DUAL and QUAD mode  |
| SD2/SCL    | D2       | Inout                   | D2 of the data bus for QUAD mode.  |
| SD3/SDA    | D3       | Inout                   | D3 of the data bus for QUAD mode.  |

**Note:** Use external 4.7 kΩ pull-up resistor.

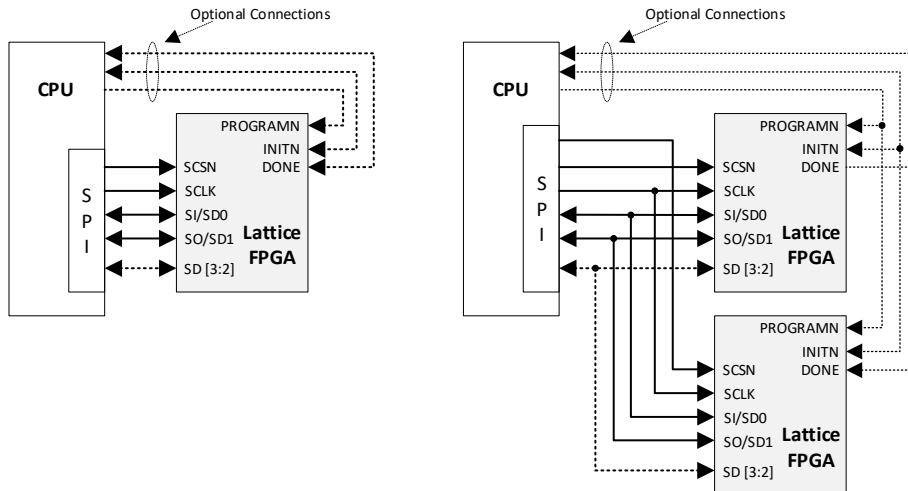


Figure 6.5. Nexus Slave SPI Port with CPU and Single or Multiple Devices

**Notes:**

- The dotted lines indicate optional connections.
- The wake up time of the device does vary with the bitstream size and the speed of the SPI port. Lattice recommends connecting the DONE pin to the CPU to monitor when the configuration is complete.

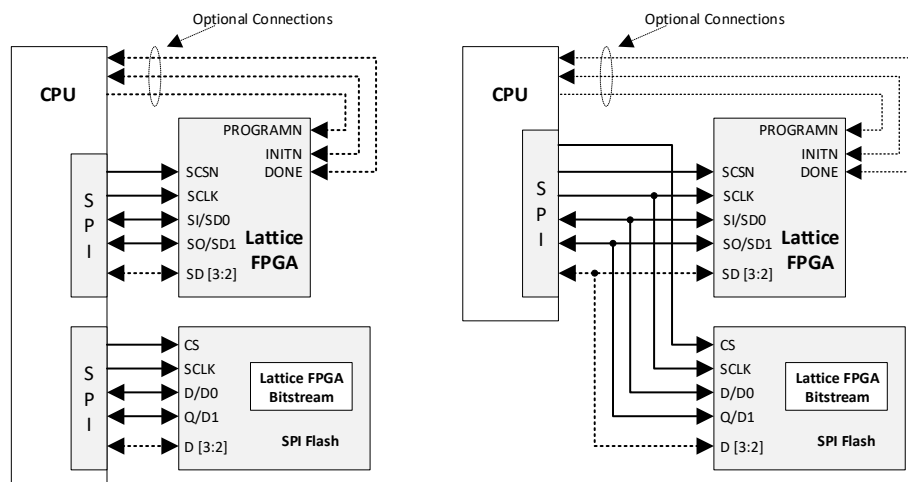


Figure 6.6. Nexus Slave SPI Port with SPI Flash

**Notes:**

- The dotted lines indicate the connection is optional.
- The Nexus device bitstream can reside in the SPI Flash device instead of the system Flash memory. The advantage of this is that the bitstream can be easily updated without changing the system software.

### 6.2.1. Method to Enable the Slave SPI Port

Similar to all configuration ports, the Slave SPI port is enabled by the two standard methods:

- Enable the Slave SPI Configuration Port in Configuration Mode.

At Power Up or PROGRAMN pin toggle LOW (for a certain period) or REFRESH command execution, holding the PROGRAMN LOW to postponed the master auto-booting event. Then drive the SCSN of the slave SPI port and shift in the Slave Configuration Port Activation Key, as shown in Table 6.4. After the Slave SPI Configuration Port is activated, the state of the PROGRAMN pin is irrelevant.

**Table 6.4. Slave SPI Configuration Port Activation Key**

| Slave Port/ Activation Key | Slave Configuration Port Activation Key |              |
|----------------------------|---|--------------|
| Slave SPI Port             | Dummy Bytes*                            | 32'HA4C6F48A |

**Note:** The number of dummy bytes should be at least 1, and only the last shifted in 32 bits matter.

- Enabling Slave SPI port persistence in user mode.

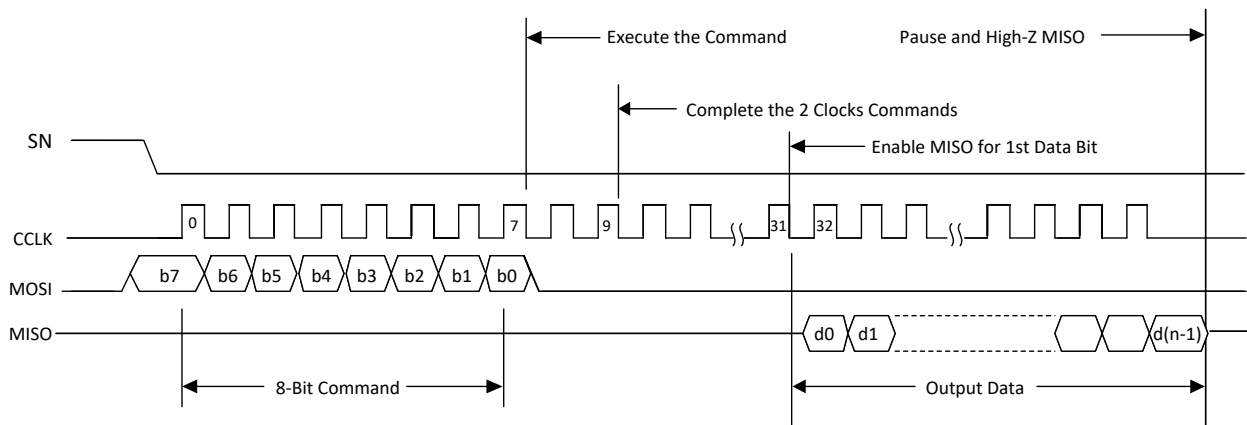
The SSPI port could be persisted in user mode by setting the desired Value (SERAL, DUAL or QUAD) for SLAVE\_SPI\_PORT in Lattice Radiant Device Constraint Editor. Once set, the configuration bitstream contains optional Slave SPI persistence bits. When the device completes configuration and wakes up, it checks the persistent bits to determine if the SLAVE SPI port is to remain operational once in user mode. This selection is independent of the CFG port arbitration during configuration phase. A port enabled by persistence is a Transparent Mode port. It reserves those pins from being occupied by user logic.

Note that both the DONE pin and the INITN pin must be high to qualify the Slave SPI port as a read back port. If not, then the device is not in user mode. The persistent bits have no affect when the device is not in user mode.

### 6.2.2. Specifications and Timing Diagrams – Slave SPI Port Waveforms

Data and commands shift into the MOSI pin on the rising edge of clock. Data is shifted out of the MISO pin on the falling edge of clock. Only a read command causes the MISO pin to be enabled for data read out.

The Slave SPI read and write waveforms are shown in Figure 6.7 and Figure 6.8.



**Figure 6.7. Slave SPI Read Waveforms**

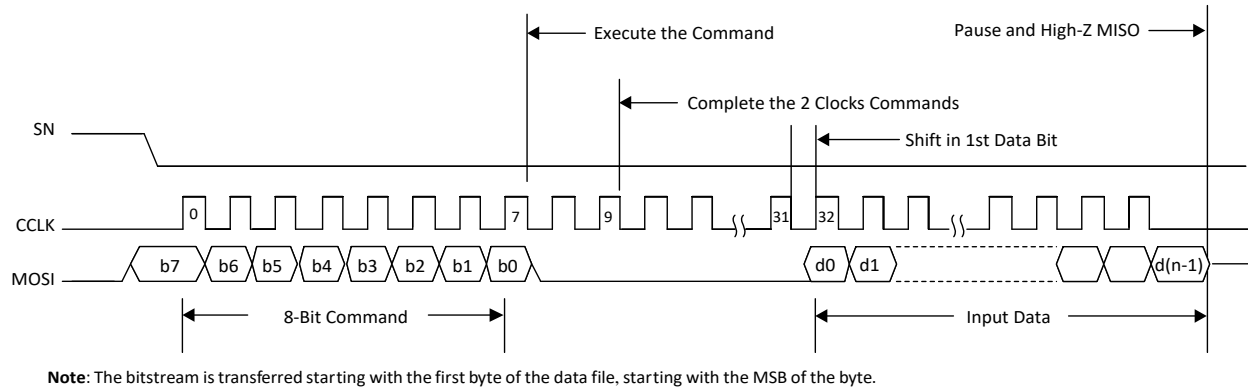


Figure 6.8. Slave SPI Write Waveforms

### 6.2.3. Slave SPI Port AC Timing Requirements

The Slave SPI port maximum operation frequency requirements is shown in [Table 6.5](#).

Table 6.5. Slave SPI port AC Timing Requirements

| Description    | Parameter  | Min | Max | Unit |
|----------------|------------|-----|-----|------|
| SCLK Frequency | $f_{CCLK}$ | —   | 135 | MHz  |

For detail AC timing requirement for the Nexus Slave SPI configuration port, refer to the AC timing section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).

### 6.2.4. Dual and Quad Slave SPI Port

By default, the SPI port data width is x1 (by 1). That is, there is only one bit of input and one bit of output. However, to allow faster loading of configuration information, some devices support wider versions of the SPI interface. For this reason, the Slave SPI configuration port of the Nexus device also support x2 (Dual) and x4 (Quad) modes of operation. For slave SPI, the Dual and Quad modes are enabled or disabled through 32-bit special commands. Those commands are processed at port level. The command codes are shown in [Table 6.6](#).

Table 6.6. Special Commands for Dual and Quad Mode Enable/Disable

| Commands              | Command Code | Note  |
|-----------------------|--------------|---|
| Dual Mode Enable      | 32'hD9B33EB3 | Enable the DUAL Mode (X2)                           |
| Quad Mode Enable      | 32'hD9B33EBD | Enable the QUAD Mode (X4)                           |
| Parallel Mode Disable | 32'hD9FFFFFF | Disable parallel mode, set back to X1 (serial) mode |

The Dual and Quad mode can only be enabled from default Serial mode. The Dual or Quad Mode Enable command has to be shift in serially from the slave SPI SI pin, and started from the first clock after the slave chip select (SCSN) pin pulled low, for 32 bits. The command execution happens upon the SCSN pulling high. Once the DUAL (X2) or QUAD (X4) mode is enabled, the Slave SPI data throughput, for either Non-JTAG command or data, is expanded from X1 to X2, X4 through the SD0 – SD3 pins as illustrated in [Table 4.4](#).

At POR or in the event of PROGRAMN pin toggle (low) or REFRESH command execution, the Slave SPI port is reset to default serial (X1) mode.

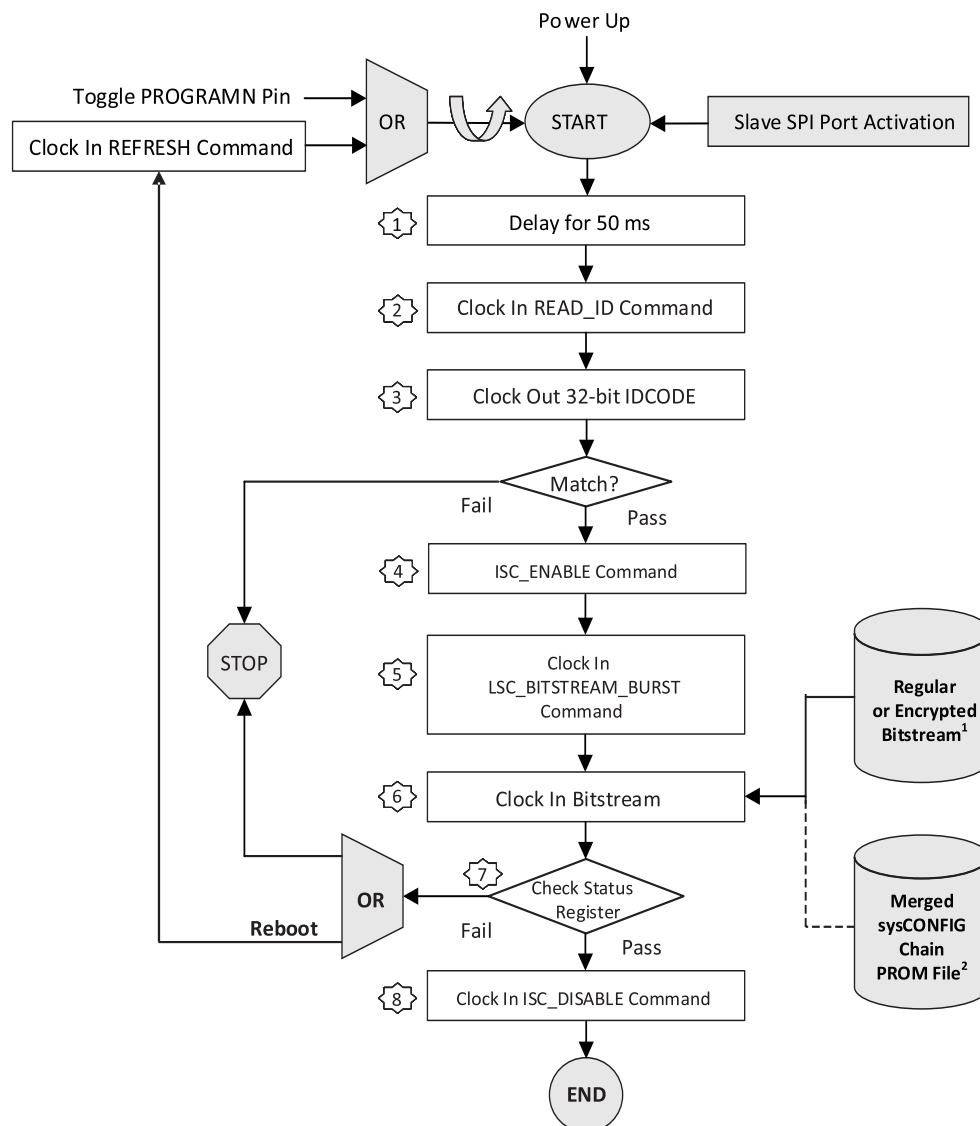


### 6.2.5. Slave SPI Configuration Flow Diagrams

The Slave SPI port supports the regular Nexus device bitstream and the encrypted bitstream (SSPI Mode).

The Slave SPI configuration flow diagram is shown in Figure 6.9. Highlights are listed below.

- The bitstream file is a stand-alone file. It is not part of the driver or system code. This provides seamless system integration and flexible file management. For example, the bitstream can be switched on the fly without changing a single line of system code.
- The Nexus device is capable to wake up the I/O X, Y to user specified value at the beginning of the bitstream.
- The Nexus FPGA Fabric wakes up and enter user mode once it reads in the entire bitstream. If it is necessary to delay the wake-up, the simplest method is by using the DONE pin. Wake-up can be delayed by holding the open-drain DONE pin low until the wake-up is desired.



**Figure 6.9. Slave SPI Configuration Flow**

**Notes:**

1. For a single Nexus device, the input file is a bitstream, which may be a standard or encrypted bitstream.
2. For a sysCONFIG chain of devices, the input file can be a merged PROM file.

## 6.2.6. Command Waveforms

### 6.2.6.1. Class A Command Waveforms

The Class A commands are ones that read data out from the Nexus device. Bit 0 of the data or bitstream is read out first. The twenty-four (24) dummy clocks provide the device the necessary delay for the proper execution of the command.

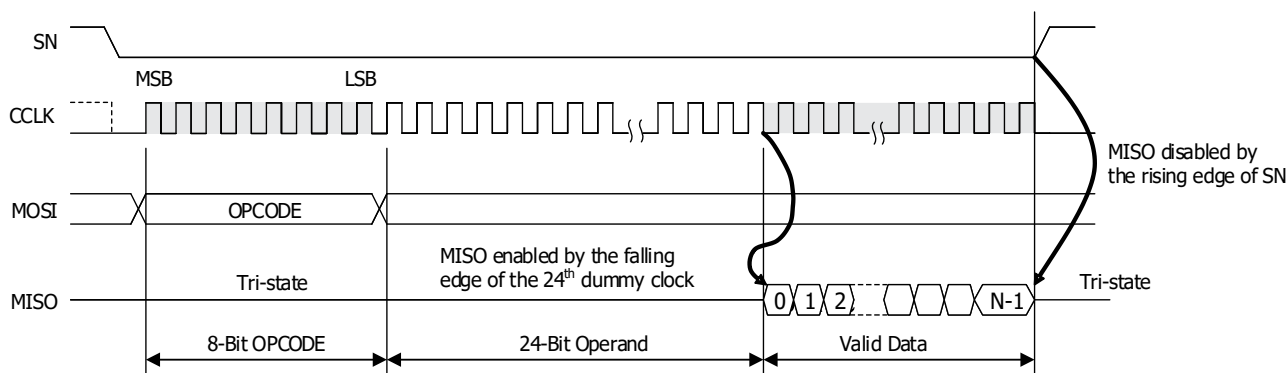


Figure 6.10. Class A Command Waveforms

### 6.2.6.2. Class B Command Waveforms

The Class B commands are used to shift data into the port. Bit 0 of the data or bitstream is shifted in first. The twenty-four (24) dummy clocks provide the device the necessary delay time to execute the command properly.

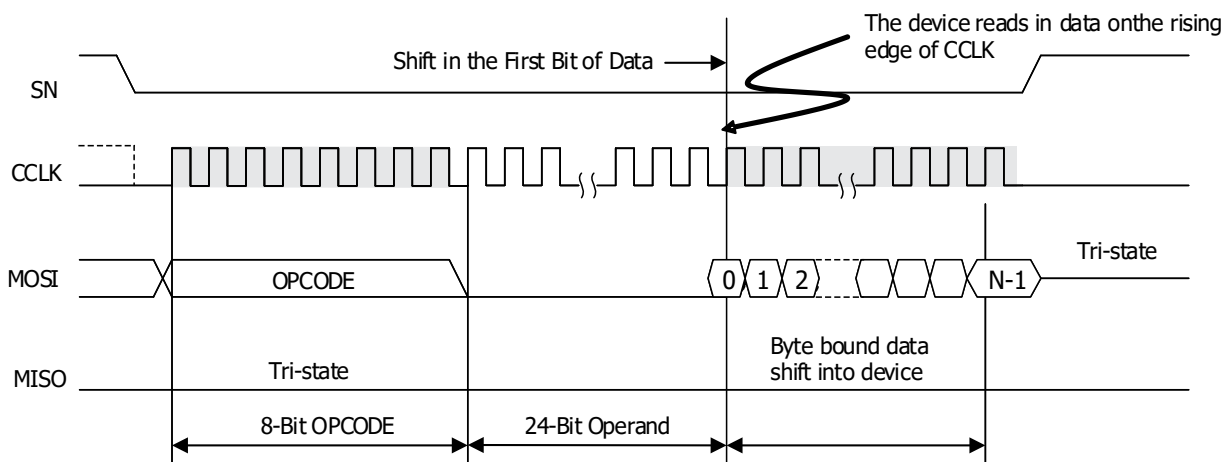


Figure 6.11. Class B Command Waveforms

### 6.2.6.3. Class C Command Waveforms

The Class C commands do not require any data to be shifted in or out. The twenty four (24) dummy clocks provide the device the necessary delay for the proper execution of the command. Even if extra dummy clocks are presented, the device ignores them.

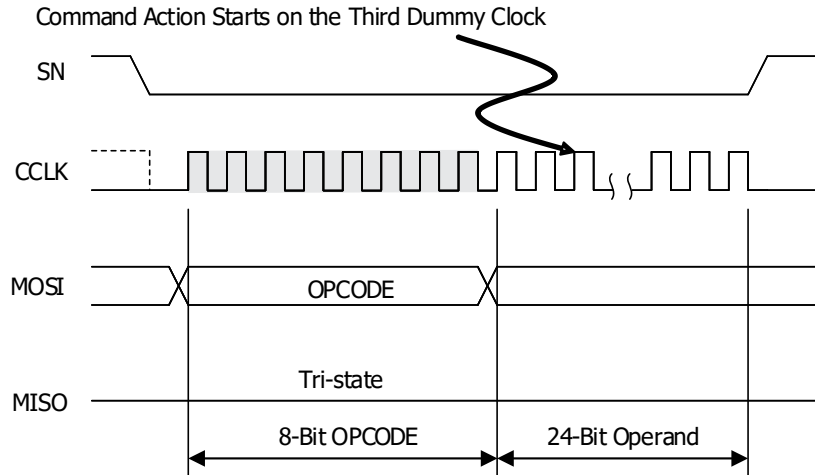


Figure 6.12. Class C Command Waveforms

### 6.2.6.4. Class D Commands Waveforms

The Class D commands do not need to shift data in or out but still require a delay to execute the action associated with the command. This type of command cannot terminate the action of any commands including itself. After the 24th dummy clock, continuing to clock or suspending the clock or driving the SN pin high does not terminate the action. The action ends when it is complete. This class of commands is defined particularly for the benefit of the two unique commands: CLEAR and REFRESH.

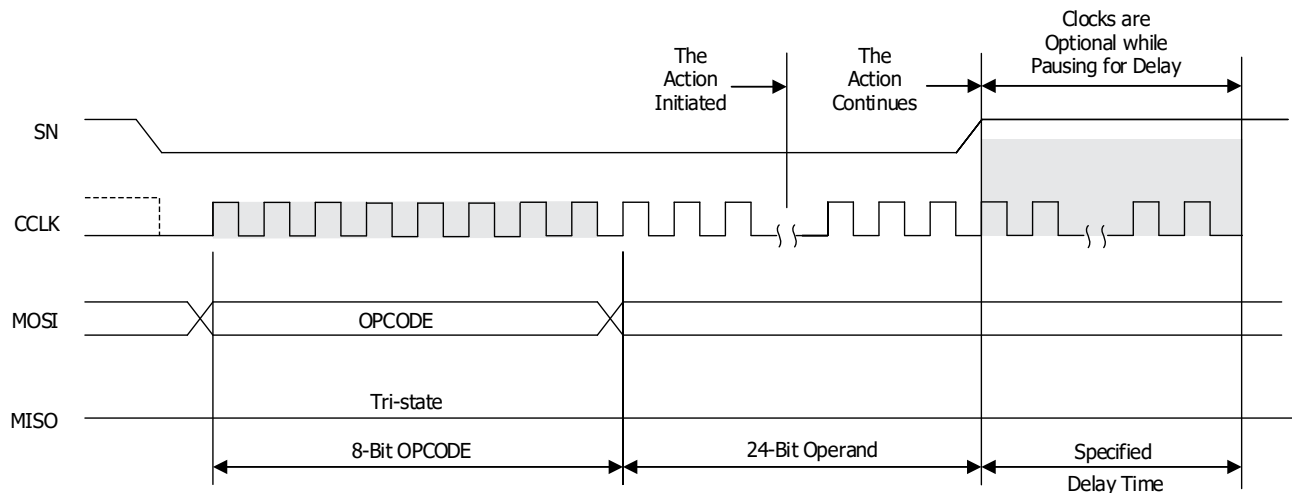


Figure 6.13. Class D Command Waveforms

### 6.2.7. Slave SPI to Master SPI Bridge

As the Master SPI and Slave SPI port on Nexus device are allocated at different set of pins, the Nexus device provides the Slave SPI to Master SPI Bridge, which allows you to program the external SPI Flash through the Nexus Slave SPI configuration port.

LSC\_PROG\_SPI command is used to enable this bridging. Once this bridging is enabled, any data following this command on SSPI interface is directed to MSPI interface. Once this access is terminated by the external Host, by deserting the Chip-Select (CSN) signal of SSPI, the bridging function is disabled and normal slave access for Configuration continues. The functional diagram of the SSPI to MSPI Bridge is shown in [Figure 6.14](#).

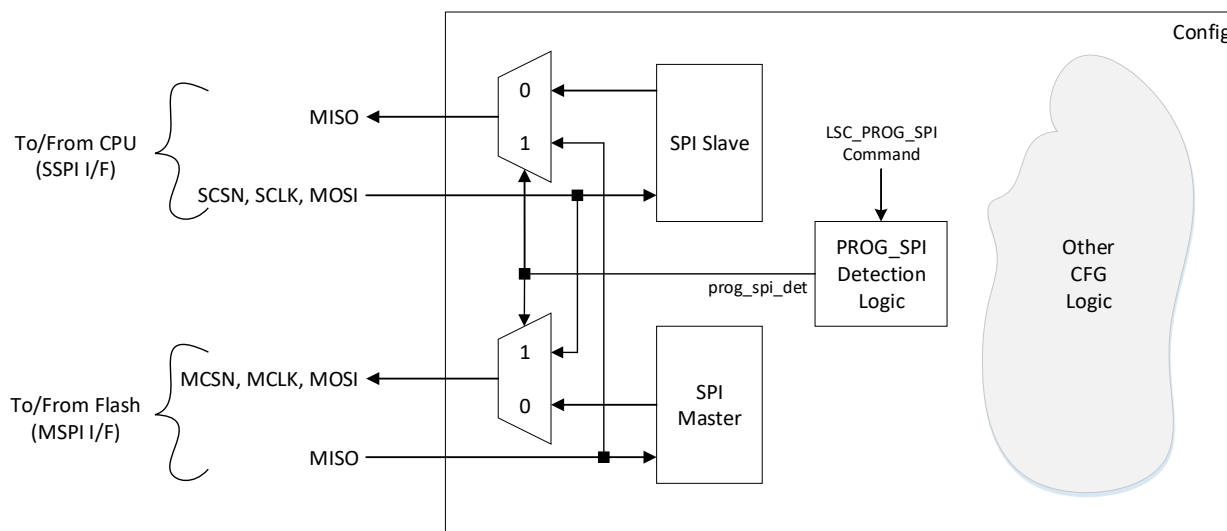


Figure 6.14. SSPI to MSPI Bridge Functional Diagram

The default support for this bridging is serial data mode only. Based on the devices I/O support, dual and quad can be supported, serial SSPI to MSPI bridging are supported by default. As an example, the diagram for SSPI to MSPI Bridge utilization is shown in [Figure 6.15](#).

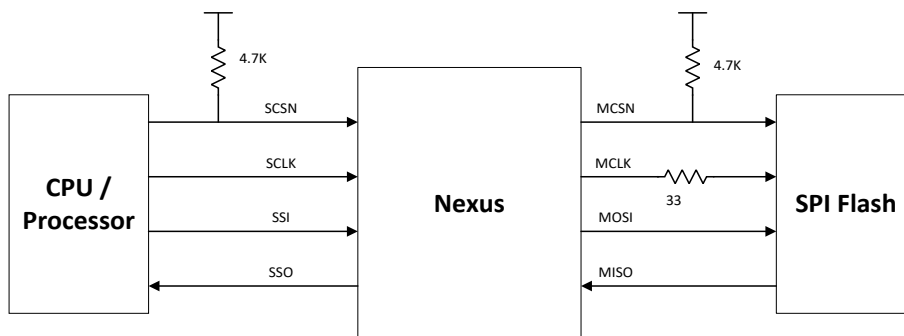


Figure 6.15. SSPI to MSPI Bridge Block Diagram

### 6.3. Slave I<sup>2</sup>C/I3C Mode

The Nexus device provides a Slave I<sup>2</sup>C/I3C configuration port that allows you to access features provided by the Configuration Logic. You can reprogram the Configuration SRAM and access status/control registers within the Configuration Logic block. When the Nexus device is in Transparent Mode, only read type commands for the Configuration SRAM are supported, allowing for device verification or debugging. The same non-JTAG command set, as shown in Table 6.7 is supported.

In the Slave I<sup>2</sup>C/I3C mode, the SD2/SCL pin becomes SCL (that is, Serial Clock) of the I<sup>2</sup>C/I3C bus, and the SD3/SDA becomes the SDA (such as Serial Data) of the I<sup>2</sup>C/I3C bus.

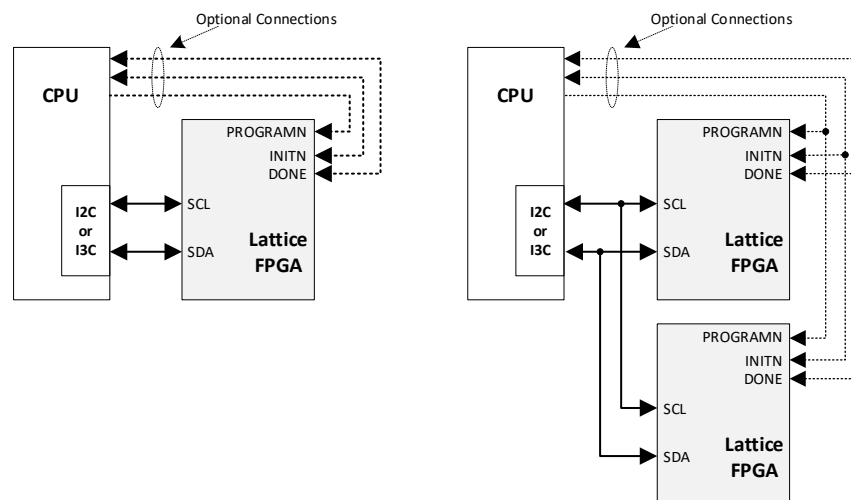
The Slave I<sup>2</sup>C/I3C port can be activated through the CFG port arbitration procedure before the device enters user mode. The I<sup>2</sup>C/I3C port can be persisted in user mode by setting the desired Value for SLAVE\_I2CI3C\_PORT in Lattice Radiant Device Constraint Editor.

Using the I<sup>2</sup>C port simplifies the Nexus device configuration process. Lattice provides C source code called i2cembedded to insulate you from the complexity of programming the Nexus device. Refer to the Lattice Radiant online help about i2cembedded.

**Table 6.7. Slave I<sup>2</sup>C/I3C Configuration Port Pins**

| Pin Name | Function | Direction | Description                           |
|----------|----------|-----------|---------------------------------------|
| SD2/SCL  | SCL      | Inout     | SCL, Serial Clock, for I2C or I3C bus |
| SD3/SDA  | SDA      | Inout     | SDA, Serial Data, for I2C or I3C bus  |

The Nexus device I<sup>2</sup>C/I3C configuration setup is shown in Figure 6.16.



**Figure 6.16. Nexus Slave I<sup>2</sup>C/I3C Port with CPU and Single or Multiple Devices**

### 6.3.1. Bus Sharing Between I<sup>2</sup>C and I3C

The Nexus device configuration block support both slave I<sup>2</sup>C and slave I3C interface, the two share the same bus as shown in Figure 6.17.

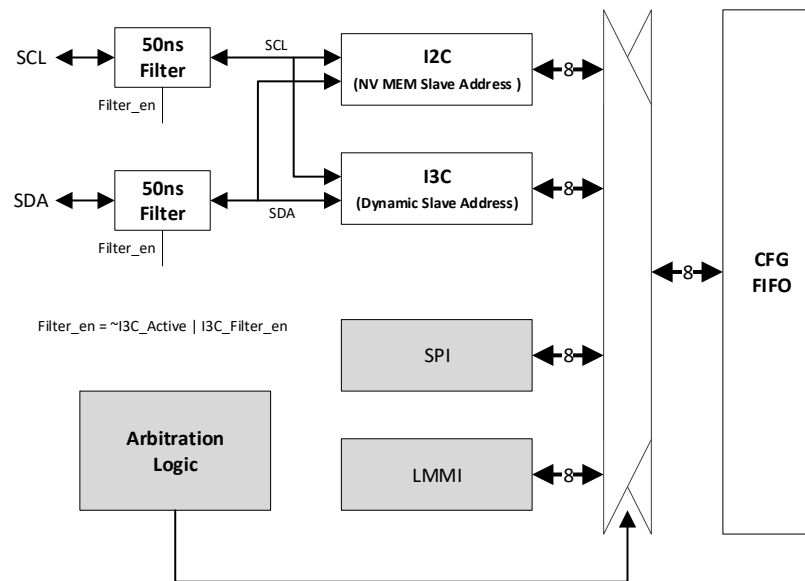


Figure 6.17. Bus Sharing between I<sup>2</sup>C and I3C

### 6.3.2. Slave I<sup>2</sup>C/I3C Configuration Mode

From configuration perspective, the I<sup>2</sup>C or I3C interface only support slave mode. They are for configuration purpose only, not usable by user logic. This hard slave I<sup>2</sup>C block and I3C block for configuration provide the industry standard two-pin interface for I<sup>2</sup>C/I3C masters to communicate with the Nexus device configuration engine.

#### 6.3.2.1. Slave I<sup>2</sup>C Port Capabilities

- Standard I<sup>2</sup>C bus protocol to communicate with configuration non-JTAG engine.
- User programmable Slave address to override the hardware default one (EFUSE, One Time Programmable).
- Built in Port Activation Detection logic to detect the Lattice Slave Configuration Port Activation Code.
- Built in PREAMBLE detection logic for bitstream alignment when execute LSC\_BITSTREAM\_BURST command.
- Built in 50 ns filter for SCL and SDA make it capable of working within newer I3C bus.

#### 6.3.2.2. Slave I3C Port Capabilities

- Two wire serial interface up to 12.5 MHz using Push-Pull.
- Legacy I<sup>2</sup>C Device co-existence on the same Bus.
- Dynamic Addressing while supporting Static Addressing for Legacy I<sup>2</sup>C support.
- Legacy I<sup>2</sup>C messaging.
- I<sup>2</sup>C -like Single Data Rate messaging (SDR).
- Built in Port Activation Detection logic to detect the Lattice Slave Configuration Port Activation Code.
- Built in PREAMBLE detection logic for bitstream alignment when execute LSC\_BITSTREAM\_BURST command.

### 6.3.3. Slave Address for I<sup>2</sup>C and I3C Ports

#### 6.3.3.1. Slave Address for I<sup>2</sup>C Configuration Port

An external I<sup>2</sup>C master accesses the Configuration Logic using address 1000000 (7-bit mode) or 1111000000 (10-bit mode) as default unless the CFG I<sup>2</sup>C base address has been modified. The CFG I<sup>2</sup>C slave address could be altered by 8 bits user programmable EFUSE bits (YYYYXXXX) inside Feature Row. Using Program File Utility – Feature Row Editor, the user specified I<sup>2</sup>C slave address becomes XXXXX00 for 7-bit mode and YYYYXXXX00 for 10-bit address.

#### 6.3.3.2. Slave Address for I3C Configuration Port

For Legacy I<sup>2</sup>C support, the Nexus device allows you to program 8 bits EFUSE bit (AAABBBBB) to setup the static address for the I3C configuration port, with BBBB00 for 7-bit mode and AAABBBBB00 for 10-bit mode. The Slave I3C Configuration Port receives its dynamic address during the Dynamic Address Assignment (DAA) process initiated by the main I3C master on the bus.

### 6.3.4. Method to Enable the Slave I<sup>2</sup>C/I3C Port

Similar to all configuration ports, the Slave I<sup>2</sup>C or I3C port is enabled by the two standard methods:

- Enable the Slave I<sup>2</sup>C Configuration Port in Configuration Mode.  
At Power Up or PROGRAMN pin toggle LOW (for certain period) or REFRESH command execution, holding the PROGRAMN LOW to postponed the master auto booting event. Then the Master performs the write activity to the slave address of the Nexus Slave I<sup>2</sup>C or I3C configuration port along with the Slave Configuration Port Activation Key, as shown in Table 6.8. After the Slave I<sup>2</sup>C or I3C Configuration Port been activated, the state of the PROGRAMN pin is irrelevant.

**Table 6.8. Slave SPI Configuration Port Activation Key**

| Slave Port/ Activation Key  | Slave Configuration Port Activation Key |              |
|-----------------------------|---|--------------|
| Slave I <sup>2</sup> C Port | Slave I <sup>2</sup> C Port Address*    | 32'HA4C6F48A |

\*Note: The slave I<sup>2</sup>C/I3C address could be either 7 bits or 10 bits address

- Enabling Slave I<sup>2</sup>C/I3C port persistence in user mode.  
The I<sup>2</sup>C/I3C port could be persisted in user mode by setting the desired Value (ENABLE) for SLAVE\_I2CI3C\_PORT in Lattice Radiant Device Constraint Editor. Once set, the configuration bitstream contains optional I<sup>2</sup>C/I2C persistence bits. When the device completes configuration and wakes up, it checks the persistent bits to determine if the SLAVE I<sup>2</sup>C/I3C port is to remain operational once in user mode. This selection is independent of the CFG port arbitration during configuration phase. A port enabled by persistence is a Transparent Mode port. It reserves those pins from being occupied by user logic.

Note that both the DONE pin and the INITN pin must be high to qualify the Slave I<sup>2</sup>C or I3C port as a read back port. If not, then the device is not in user mode. The persistent bits have no affect when the device is not in user mode.

#### 6.3.4.1. Slave I<sup>2</sup>C Port Activation Flow

The standard Slave I<sup>2</sup>C Configuration Port activation flow is shown in Figure 6.18.

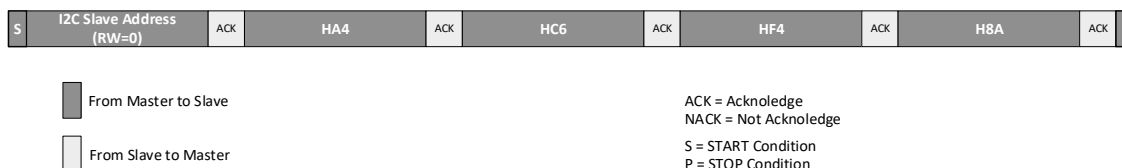


Figure 6.18. Slave I<sup>2</sup>C Configuration Activation Flow

#### 6.3.4.2. Slave I3C Port Activation Flow

The typical Slave I3C Configuration Port activation flow is shown in Figure 6.19.

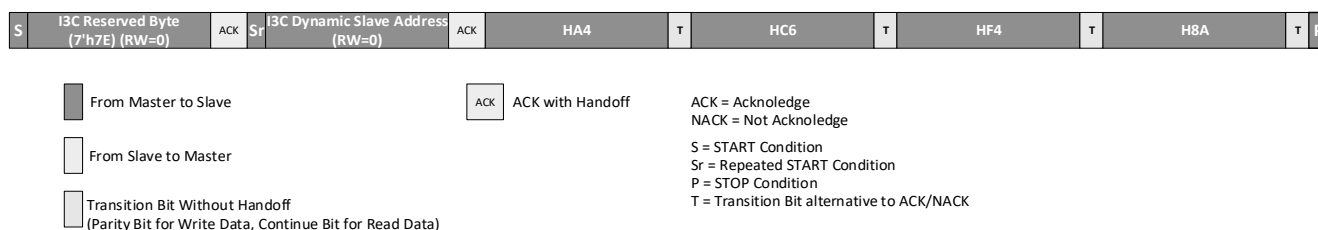


Figure 6.19. Slave I3C Configuration Activation Flow

#### 6.3.5. Slave I<sup>2</sup>C C/I3C Configuration Logic Access

The Slave I<sup>2</sup>C Configuration Port or Slave I3C Configuration Port follows corresponding standard to access the Nexus device non-JTAG configuration engine, using the same non-JTAG command set, as shown in Table 6.14.

##### 6.3.5.1. Slave I<sup>2</sup>C Configuration Access Flow

The typical I<sup>2</sup>C configuration write and read flow are shown in Figure 6.20 and Figure 6.21.

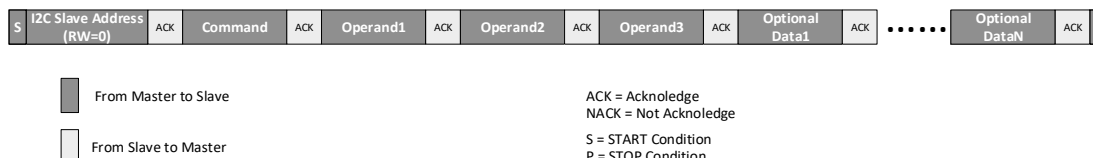


Figure 6.20. Typical I<sup>2</sup>C Write

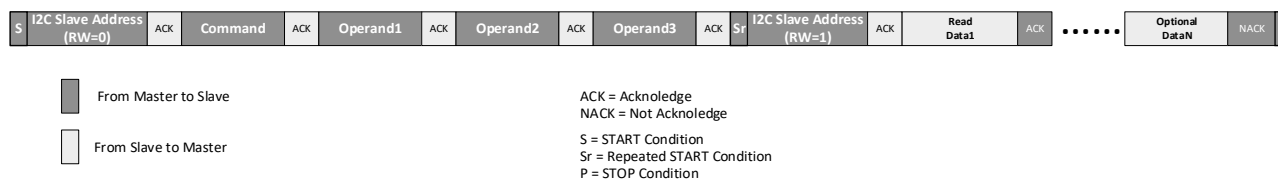


Figure 6.21. Typical I<sup>2</sup>C Read



### 6.3.5.2. Slave I3C Configuration Access Flow

The typical I3C configuration write and read flow are shown in Figure 6.22, Figure 6.23, Figure 6.24, and Figure 6.25.

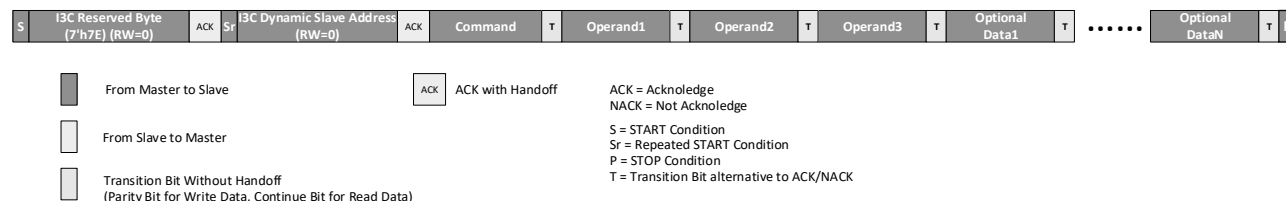


Figure 6.22. Typical I3C Write with 7'h7E

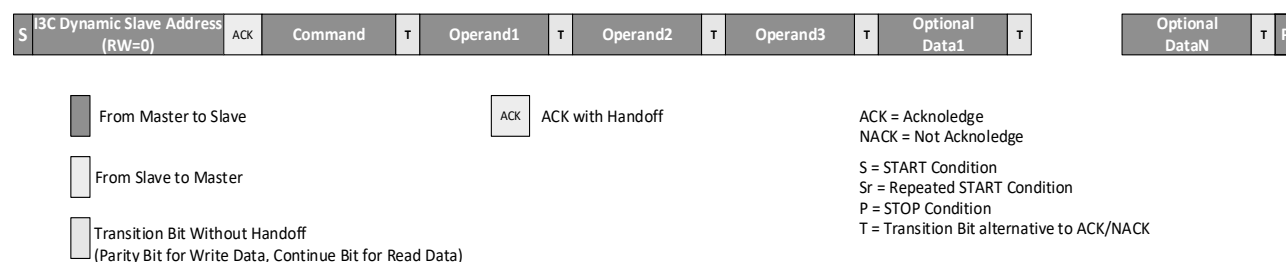


Figure 6.23. Typical I3C Write without 7'h7E

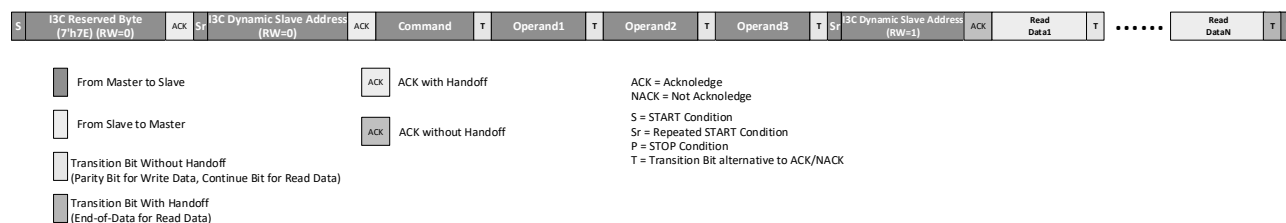


Figure 6.24. Typical I3C Read with 7'h7E

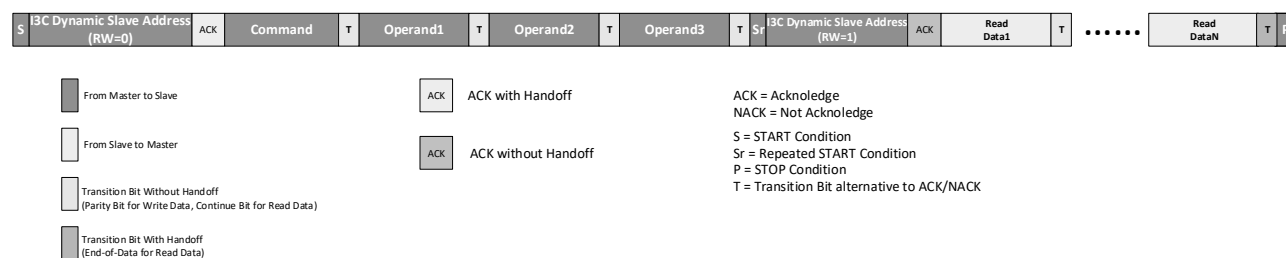


Figure 6.25. Typical I3C Read without 7'h7E

### 6.3.6. Slave I<sup>2</sup>C/I<sup>3</sup>C AC Timing Requirements

The Slave I<sup>2</sup>C and I<sup>3</sup>C maximum operation frequency requirements are listed in [Table 6.10](#).

**Table 6.9. Slave I<sup>2</sup>C/I<sup>3</sup>C Maximum Frequency Requirements**

| Description                                  | Parameter            | Min | Max | Unit |
|--|----------------------|-----|-----|------|
| I <sup>2</sup> C Maximum SCL Clock Frequency | F <sub>SCL_I2C</sub> | —   | 1   | MHz  |
| I <sup>3</sup> C Maximum SCL Clock Frequency | F <sub>SCL_I3C</sub> | —   | 12  | MHz  |

For detail AC timing requirement for the Nexus device Slave I<sup>2</sup>C/I<sup>3</sup>C configuration port, refer to the AC timing section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Curtis-NX Family Data Sheet \(FPGA-DS-02078\)](#).

## 6.4. JTAG Mode

The Nexus device provides a four-pin JTAG engine, which is fully compliance with IEEE 1149 (Standard Test Access Port and Boundary-Scan Architecture) and IEEE1532 (Standard for In-System Configuration of Programmable Devices) standards. A separate dedicated JTAG\_ENABLE pin can enable the JTAG port at any time. The JTAG port on Nexus devices provides:

- Offline external Flash memory programming
- Background external Flash memory programming
- Direct SRAM configuration
- Full access to the Nexus Device Configuration Logic
- Device chaining
- IEEE 1149.1 testability
- IEEE 1532 Compliant programming

The advantages of keeping the JTAG port available include:

- Multi-chain Architectures – The JTAG port is the only configuration and programming port that permits the Nexus device to be combined in a chain of other programmable logic.
- Reveal Debug – The Lattice Reveal debug tool is an embeddable logic analyzer tool. It allows you to analyze the logic inside the Nexus device in the same fashion as an external logic analyzer permits analysis of board level logic. Reveal access is only available via the JTAG port.
- SRAM Readback – The JTAG port is able to directly access the configuration SRAM. It is occasionally necessary to perform failure analysis for SRAM based FPGAs. A key component to failure analysis can involve reading the configuration SRAM.
- Boundary Scan Testability – Board level connectivity testing performed using IEEE 1149.1 JTAG is a key capability for assuring the quality of assembled printed-circuit-boards. Lattice provides Boundary Scan Description Language files for the Nexus device on the Lattice website.

### 6.4.1. Method to Enable the JTAG port

The four pins for the JTAG port on Nexus devices are dual-purpose I/O, which are shared with user functionalities. Once the dedicated JTAG-ENABLE pin is driven HIGH, it can enable the JTAG port at any time, no matter whether the device is in configuration mode or user functional mode.

### 6.4.2. JTAG Port AC Timing Requirements

The JTAG port AC timing requirements are listed in [Table 6.10](#).

**Table 6.10. JTAG AC Timing Requirements**

| Description   | Parameter | Min | Max | Unit |
|---------------|-----------|-----|-----|------|
| TCK Frequency | $f_{MAX}$ | —   | 25  | MHz  |

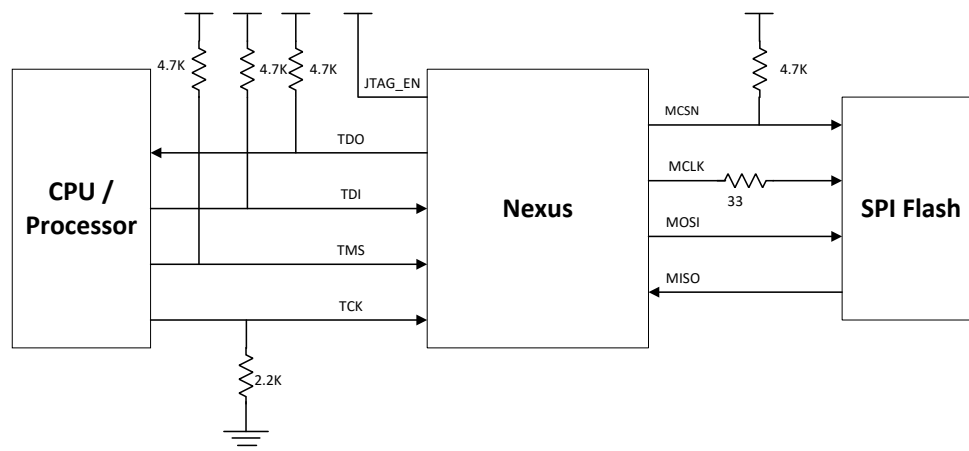
For detail AC timing requirement for the Nexus JTAG port, refer to the AC timing section of [CrossLink-NX Family Data Sheet \(FPGA-DS-02049\)](#) and [Certus-NX Family Data Sheet \(FPGA-DS-02078\)](#).

### 6.4.3. JTAG to Master SPI Bridge

The Nexus device provides the JTAG to Master SPI Bridge, which allows you to program the external SPI Flash through the Nexus JTAG port.

LSC\_PROG\_SPI instruction is used to enable this bridging. There is a 16 bit TDR between TDI and TDO for this instruction. Upon the TAP controller going through the Update-DR state first time, the internal TDI signal, the gated TCK signal, the Shift-DR signal, and the internal TDO signal are unconditionally connected respectively to the MOSI/MD0 pin, the MCLK pin, the MCSN pin, and the MISO/MD1 pin.

As an example, the diagram for JTAG to MSPI Bridge utilization is shown in [Figure 6.26](#).

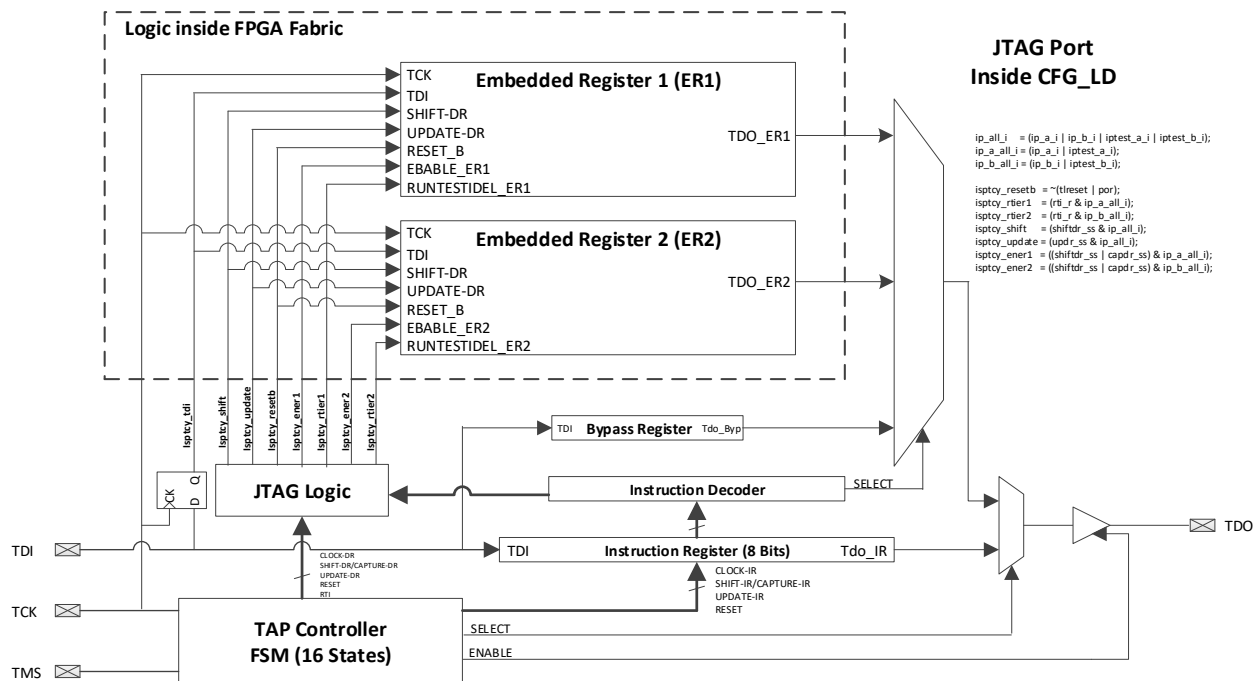


**Figure 6.26. JTAG to MSPI Bridge Block Diagram**

### 6.4.4. JTAG ispTracy/Reveal Support

The JTAG ispTracy feature supported in FPGAs is the optional addition of internal logic analyzers (ILA). These ILAs have similar features to external logic analyzers such as programmable event and trigger conditions and deep trace memory. Logic analyzer IP (two current versions are currently supported: ispTRACY and REVEAL) consists of one ispJTAG core plus numerous ispLA logic analyzer cores (device limited). The IP block ispJTAG implements two embedded registers to support logic analyzers. To implement the ispJTAG core, the configuration block has an 11-port I/O interface to the core logic as seen in [Figure 6.27](#).

In order to be IEEE 1149.1 compliant, there is a default 16-bit register implemented for registers ER1 and ER2 in order to allow the size of this register to be a fixed length whenever these registers are not implemented in FPGA logic. Whenever these registers are actually implemented in FPGA logic, these 16-bit registers are replaced with a new fixed length register that is based on the customer application – in order to allow for 1149.1 compliance, the BSDL file needs to be modified by the customer to define these new ER1 and/or ER2 register lengths. This new implementation is different from that of all previous device architectures.



**Figure 6.27. JTAG ispTracy Interface**

In user functional mode, the four JTAG pins located in I/O BANK1 can have their functionality changed from user function to JTAG function by bringing JTAGEN pin to VCCIO1. Hence, the TMS/TDI/TCK pins need to be configured as INPUT or BI-DIRECTIONAL in user mode.

The JTAG driver to Nexus needs to match the JTAG pins I/O type. For example, if 4 JTAG pins in Bank 1 use LVCMOS33 then the JTAG driver needs to be 3.3 V.

When JTAGEN is high in user mode, the four signals need to be HIGHZ from other devices beside the JTAG driver.

After the JTAGEN goes low in user mode, the drive strength of these four pins remain 8 mA until 200 ms later and need dynamic switching to return to user mode drive strength. If these four pins stay static after JTAGEN goes low, they remain 8 mA drive strength until switching.

## 6.5. Device Status Register

The Nexus device has a 64-bit device status register, which indicates the status of the device. The READ\_STATUS command shifts out this 64-bit internal status of the device. The device status register can only be read by the slave configuration port, such as JTAG port, Slave SPI port, Slave I<sup>2</sup>C port, Slave I3C port or LMMI interface.

Table 6.11. 32-Bit Device Status Register

| Bit   | Function                | Status Value |              |          | Comments  |
|-------|-------------------------|--------------|--------------|----------|---|
|       |                         | Reset Value  | 0            | 1        |   |
| 0     | Transparent Mode        | 0            | No           | Yes      | Device is currently in Transparent mode   |
| [3:1] | Config Target Selection | 000          | —            | —        | 000 SRAM array  |
|       |                         |              |              |          | 001 EFUSE Normal;<br>Program input data to EFUSE Memory directly,<br>Read back data from Shadow register.   |
|       |                         |              |              |          | 010 EFUSE Pseudo;<br>Program input data to Shadow Register,<br>Read back data from Shadow register.   |
|       |                         |              |              |          | 011 EFUSE Safe;<br>Program Shadow Register data into EFUSE<br>memory, Read back data from Shadow<br>register.   |
| 4     | JTAG Active             | 0            | No           | Yes      | JTAG ISC Machine is currently active  |
| 5     | PWD Protection          | 0            | No           | Yes      | Configuration logic is password protected   |
| 6     | OTP                     | 0            | No           | Yes      | NV User Feature Sector OTP is Set   |
| 7     | Reserved                | 0            | —            | —        | —   |
| 8     | DONE                    | 0            | No           | Yes      | Done bit has been set   |
| 9     | ISC Enable              | 0            | No           | Yes      | JTAG instructions are being executed with ISC Enabled   |
| 10    | Write Enable            | 0            | Not Writable | Writable | Selected configuration target is write-protected from at least one of the following setup:<br><ul style="list-style-type: none"> <li>Selected configuration target security bit is set</li> </ul> Password protection is enabled and password is mismatch |
| 11    | Read Enable             | 0            | Not Readable | Readable | Read-protected from at least one of the following setup:<br><ul style="list-style-type: none"> <li>Security bit is set</li> </ul> Password protection is enabled and password is mismatch   |
| 12    | Busy Flag               | 0            | No           | Yes      | Configuration logic is busy   |
| 13    | Fail Flag               | 0            | No           | Yes      | Last instruction/command execution failed   |
| 14    | Reserved                | 0            | —            | —        | —   |
| 15    | Decrypt Only            | 0            | No           | Yes      | Only encrypted data are accepted  |
| 16    | PWD Enable              | 0            | No           | Yes      | Password protection is enabled for EFUSE  |
| 17    | PWD All                 | 0            | No           | Yes      | Password protection is extended for SRAM  |
| 18    | CID EN                  | 0            | No           | Yes      | Customer ID is enabled for IDCODE_PUB command   |
| 19    | Internal use            | 0            | —            | —        | Not used  |
| 20    | Reserved                | 0            | —            | —        | —   |
| 21    | Encrypt Preamble        | 0            | No           | Yes      | Encrypted Preamble is detected  |
| 22    | STD Preamble            | 0            | No           | Yes      | Standard Preamble is detected   |
| 23    | SPIm Fail 1             | 0            | No           | Yes      | Failed to configure from the primary pattern  |

| Bit     | Function  | Status Value |     |      | Comments  |
|---------|---|--------------|-----|------|---|
|         |   | Reset Value  | 0   | 1    |   |
| [27:24] | BSE Error Code                                    | 0000         | —   | —    | 0000 No error   |
|         |   |              |     |      | 0001 ID error   |
|         |   |              |     |      | 0010 CMD error - illegal command detected   |
|         |   |              |     |      | 0011 CRC error  |
|         |   |              |     |      | 0100 PRMB error - preamble error  |
|         |   |              |     |      | 0101 ABRT error - configuration is aborted  |
|         |   |              |     |      | 0110 OVFL error - data overflow error   |
|         |   |              |     |      | 0111 SDM error - bitstream pass the size of the SRAM array  |
|         |   |              |     |      | 1000 Authentication Error*  |
|         |   |              |     |      | 1001 Authentication Setup Error*  |
|         |   |              |     |      | 1010 Bitstream Engine Timeout Error   |
| 28      | Execution Error                                   | 0            | No  | Yes  | Error occur during execution.   |
| 29      | ID Error  | 0            | No  | Yes  | ID mismatch with Verify_ID command  |
| 30      | Invalid Command                                   | 0            | No  | Yes  | Invalid command received  |
| 31      | WDT Busy  | 0            | No  | Yes  | Watch Dog Timer is busy   |
| 32      | Reserved  | 0            | —   | —    | —   |
| 33      | Dry Run DONE                                      | 0            | No  | Yes  | Bit to indicate the pseudo done bit been set by Dry Run activity                                      |
| [37:34] | BSE Error 1 Code for previous bitstream execution | 0000         | —   | —    | 0000 No error   |
|         |   |              |     |      | 0001 ID error   |
|         |   |              |     |      | 0010 CMD error - illegal command detected   |
|         |   |              |     |      | 0011 CRC error  |
|         |   |              |     |      | 0100 PRMB error - preamble error  |
|         |   |              |     |      | 0101 ABRT error - configuration is aborted  |
|         |   |              |     |      | 0110 OVFL error - data overflow error   |
|         |   |              |     |      | 0111 SDM error - bitstream pass the size of the SRAM array  |
|         |   |              |     |      | 1000 Authentication Error*  |
|         |   |              |     |      | 1001 Authentication Setup Error*  |
|         |   |              |     |      | 1010 Bitstream Engine Timeout Error   |
| 38      | Bypass Mode                                       | 0            | No  | Yes  | Device currently in Bypass mode   |
| 39      | Flow Through Mode                                 | 0            | No  | Yes  | Device currently in Flow Through Mode   |
| 40      | Reserved  | 0            | —   | —    | —   |
| 41      | Reserved  | 0            | —   | —    | —   |
| 42      | SFDP Timeout                                      | 0            | No  | Yes  | Indicates MSPI time out when trying to read the signature from flash                                  |
| 43      | Key Destroy Pass <sup>1</sup>                     | 0            | No  | Yes  | Key destroy function passed   |
| 44      | INITN   | 0            | Low | High | INITN Pin state.  |
| 45      | I3C Parity Error 2                                | 0            | No  | Yes  | I3C received a parity error type S2   |
| 46      | INIT Bus ID Error                                 | 0            | No  | Yes  | Initialization Bus ID Error occurred  |
| 47      | I3C Parity Error 1                                | 0            | No  | Yes  | I3C received a parity error type S1   |
| 49:48   | Authentication Mode*                              | 0            | —   | —    | Indicates what authentication mode is set.<br>00 - No Auth<br>01 - ECDSA<br>10 - HMAC<br>11 - No Auth |
| 50      | Authentication Done*                              | 0            | No  | Yes  | Indicates that authentication has completed.  |

| Bit | Function                                 | Status Value |    |     | Comments  |
|-----|--|--------------|----|-----|---|
|     |  | Reset Value  | 0  | 1   |   |
| 51  | Dry Run Authentication Done <sup>1</sup> | 0            | No | Yes | Indicates that Dry Run authentication has completed.                |
| 52  | JTAG Locked                              | 0            | No | Yes | Indicates that the JTAG port is locked                              |
| 53  | SSPI Locked                              | 0            | No | Yes | Indicates that the Slave SPI port is locked                         |
| 54  | I <sup>2</sup> C/I <sup>3</sup> C Locked | 0            | No | Yes | Indicates that the I <sup>2</sup> C/I <sup>3</sup> C port is locked |
| 55  | PUB Read Lock <sup>1</sup>               | 0            | No | Yes | ECDSA Public key read lock is enabled.                              |
| 56  | PUB Write Lock <sup>1</sup>              | 0            | No | Yes | ECDSA Public key write lock is enabled.                             |
| 57  | FEA Read Lock                            | 0            | No | Yes | Feature Row Read Lock is enabled                                    |
| 58  | FEA Write Lock                           | 0            | No | Yes | Feature Row Write Lock is enabled                                   |
| 59  | AES Read Lock                            | 0            | No | Yes | AES Key Read Lock is enabled  |
| 60  | AES Write Lock                           | 0            | No | Yes | AES Key Write Lock is enabled                                       |
| 61  | PWD Read Lock                            | 0            | No | Yes | Password Read Lock is enabled                                       |
| 62  | PWD Write Lock                           | 0            | No | Yes | Password Write Lock is enabled                                      |
| 63  | Global Lock                              | 0            | No | Yes | Global Lock is enabled for OTP rows                                 |

<sup>1</sup>Note: For Certus-NX devices only. Treat as RESERVED for CrossLink-NX.

## 6.6. Control Register 0 (CR0)

The 32 bits Control Register 0 is used to control configuration logic behavior during and after configuration. Write the CR0 through LSC\_PROG\_CNTRL0 command and read back the CR0 content through LSC\_READ\_CNTRL0 command. Also, the CR0 could be written through the LSC\_PROG\_CNTRL0 command with the Bitstream file. The bit definition of the CR0 is shown in Table 6.12.

**Table 6.12. Bit Definition for Control Register 0**

| BIT     | Description  | Default | Note   |
|---------|--------------|---------|--|
| [31:30] | Reserved     | 0       | —  |
| 29      | WKUP TRAN    | 0       | Control bit transparent reconfiguration for the event of PROGRAMN pin toggle or REFRESH command execution. When this bit is set to 0, the wake-up signals GOE, GWE and GSRN goes low during reconfiguration. When this bit is set to 1, the wake-up signals GOE, GWE and GSRN are set according user setting of NDR when PROGRAMN pin toggle or REFRESH command execution. |
| 28      | NDR(TransFR) | 0       | Non-Disturbing Re-configure: Control bit for supporting Leave Alone I/O for reconfiguration. If set, the I/O output maintains previous value after entering ISC ACCESS state for configuration instead being tristated when device enters transparent access mode from PROGRAMN pin toggle or REFRESH command execution.   |
| [27:26] | Reserved     | 0       | —  |
| 25      | Tran CRAM    | 0       | Control bit to enable the write operation to the SRAM array in transparent mode.   |
| 24      | Tran EBR     | 0       | This bit provides user option for EBR control through the Init Bus in transparent access mode. If set, configuration logic takes over the control of EBR in transparent access mode, user access to EBR is suspended.  |
| 23      | Tran IP      | 0       | This bit provides user option for Hard IP control in transparent access mode. If set, configuration logic takes over the control of the Initialization Bus in transparent access mode, user access to Hard IPs is suspended.   |
| [22:20] | RESERVED     | 0       | —  |
| 19      | SPIM         | 0       | Control bit for Multiple Boot Enable for next refresh event (PROGRAMN pin toggle or REFRESH command execution);<br>1 – Use boot address from CIB SRAM bits depend bitstream setting.<br>0 – Use hard coded boot address (H000000 for first boot; HFFFF00 for second boot)  |

| BIT     | Description                               | Default | Note  |
|---------|---|---------|---|
| [18:17] | P_DONE CTRL                               | 0       | Overload control for PROGRAM_DONE command in bitstream for configuration daisy chain setup:<br>10 – Overload with BYPASS<br>11 – Overload with FLOW_THROUGH<br>0X – No Overload (Default) |
| [16:8]  | Reserved                                  | 0       | —   |
| [7:6]   | Slew rate control for Config output pins. | 0       | Slew rate control for configuration output pins defined as:<br>00 – Slow<br>01 – Medium<br>10 – Fast<br>11 – Fast   |
| [5:0]   | Master SPI Clock Select                   | 0       | Control bits that set the divide ratio to derive the Master SPI clock from the on-chip oscillator.  |

## 6.7. Control Register 1 (CR1)

The 32 bits Control Register 1 is used to control EFUSE/OTP access control logic behavior during and after configuration. User could Write the CR1 through LSC\_PROG\_CNTRL1 command and read back the CR1 content through LSC\_READ\_CNTRL1 command. The CR1 can also be written through the LSC\_PROG\_CNTRL1 command with the Bitstream file. The bit definition of the CR1 is shown in [Table 6.13](#).

**Table 6.13. Bit Definition for Control Register 1**

| Bit     | Definition                        | Default | Note   |
|---------|-----------------------------------|---------|--|
| [31:24] | Reserved                          | 0       | —  |
| 23      | CPHA                              | 0       | This Control bit is used to select SPI clock format.<br>0 – Sampling of data occurs at the rising edge of the clock<br>1 – Sampling of data occurs at the falling edge of the clock  |
| 22      | CPOL                              | 0       | This control bit selects an inverted or non-inverted clock<br>0 – Active high clocks selected. In idle state clock is low<br>1 – Active low clocks selected. In idle state clock is high   |
| [21:18] | Reserved                          | 0       | —  |
| 17      | 32-bit SPIM address               | 0       | Enable 32-bit MSPI addressing mode, default 24-bit addressing.   |
| 16      | Reserved                          | 0       | —  |
| 15      | SFDP Enable                       | 0       | Enables checking the SFDP signature during master SPI booting. Should be enabled for flash devices that support SFDP.<br>1 – Enable reading SFDP signature from flash SFDP header.<br>0 – Enable reading LSCC signature from flash starting at address 0x000 |
| 14      | 32-bit SPIM commands              | 0       | Send 32-bit command set on MSPI, default 24-bit command set.   |
| 13      | Disable I/O glitch filter         | 0       | Disable I/O Glitch Filter during config  |
| 12      | Reserved                          | 0       | —  |
| [11:8]  | Reserved                          | 0000    | —  |
| [7:5]   | Master Preamble Timer count value | 000     | preamble_count: number of clock cycles to get a valid preamble before declare error.<br>000 -> 600000<br>001 -> 400000<br>010 -> 200000<br>011 -> 40000<br>100 -> 20000<br>101 -> 4000<br>110 -> 2000<br>111 -> 400  |



| Bit   | Definition                        | Default | Note  |
|-------|-----------------------------------|---------|---|
| 4     | Signature_dis                     | 0       | If FSDP Enable = 0 do not read LSCC signature from flash, go directly to reading preamble.<br>1 – Skip checking signature when FSDP Enable = 0;<br>0 – Signature Enabled (default 0). |
| [3:2] | Master Preamble Timer retry value | 0       | Number of times the Master Preamble detection should be retried.<br>00 – No retry<br>01 – Retry 1 time<br>10 – Retry 3 times<br>11 – Reserved   |
| 1     | Reserved                          | 0       | —   |
| 0     | I/O Ready Disable                 | 0       | When set to 1 disables waiting for I/O ready for device wakeup.   |

## 6.8. TransFR Operation

The Nexus, like other Lattice FPGAs, provides for the TransFR™ capability. TransFR is described in [TransFR Usage Guide for Nexus Platform \(FPGA-TN-02173\)](#).

## 6.9. SPI/I<sup>2</sup>C/I3C Command Support

The ISC and programming commands for all non-JTAG configuration interface such as SPI, I<sup>2</sup>C, and I3C are shown in [Table 6.14](#).

**Table 6.14. Slave Configuration Interface Command Table**

| Command            | Binary   | Hex | Operand | Class <sup>4</sup> | Flow Operation Number <sup>5</sup> | Delay Time   | Description   |
|--------------------|----------|-----|---------|--------------------|------------------------------------|--------------|---|
| ISC_NOOP           | 11111111 | FF  | 0 bits  | C                  | —                                  | None         | Non-operation   |
| ISC_NOOP0          | 00000000 | 00  | 0 bits  | C                  | —                                  | None         | Non-operation   |
| READ_ID            | 11100000 | E0  | 24 bits | A                  | 2, 3                               | None         | Read out the 32-bit IDCODE of the device  |
| READ_UNIQUEID      | 00011001 | 19  | 24 bits | A                  | —                                  | None         | Reads the 64-bit UNIQUE ID, 56 bits set by Lattice, 8 bits set by customer.   |
| USERCODE           | 11000000 | C0  | 24 bits | A                  | —                                  | None         | Read 32-bit usercode  |
| LSC_READ_STATUS    | 00111100 | 3C  | 24 bits | A                  | 7                                  | None         | Read out internal status  |
| LSC_CHECK_BUSY     | 11110000 | F0  | 24 bits | A                  | —                                  | None         | Read 1 bit busy flag to check the command execution status  |
| LSC_DEVICE_CONTROL | 01111101 | 7D  | 24 bits | B                  | —                                  | <sup>3</sup> | An 8-bit field in Operand 1 allows various device control, such as GSR, Sleep, Dry Run, and others.<br>Bit 0: Cause a Global Set/Reset to occur on the device.<br>Bit 1: Launch a One-time check of SED.<br>Bit 2: Reserved<br>Bit 3: Cause a reset on configuration logic related FSM and flags.<br>Bit 4: Reserved for internal dry-run.<br>Bit [6:5]: Launch Dry-Run event.<br>2'b01 – Dry-Run for primary bitstream |

| Command              | Binary   | Hex | Operand | Class <sup>4</sup> | Flow Operation Number <sup>5</sup> | Delay Time   | Description  |
|----------------------|----------|-----|---------|--------------------|------------------------------------|--------------|--|
|                      |          |     |         |                    |                                    |              | 2'b10 – Dry-Run for secondary bitstream<br>2'b11 – Slave Dry-Run<br>2'b00 - no Dry-Run<br>Bit 7: CRC check bit                   |
| LSC_REFRESH          | 01111001 | 79  | 24 bits | D                  | —                                  | <sup>3</sup> | Equivalent to toggle PROGRAMN pin  |
| ISC_ENABLE           | 11000110 | C6  | 24 bits | C                  | 4                                  | None         | Enable the Offline configuration mode  |
| ISC_ENABLE_X         | 01110100 | 74  | 24 bits | C                  | —                                  | None         | Enable the Transparent configuration mode  |
| ISC_DISABLE          | 00100110 | 26  | 24 bits | C                  | —                                  | None         | Disable the configuration operation  |
| LSC_BITSTREAM_BURST  | 01111010 | 7A  | 24 bits | B                  | 8                                  | <sup>3</sup> | Configure the Device by burst in bitstream file in slave mode  |
| ISC_PROGRAM_USERCODE | 11000010 | C2  | 24 bits | B                  | —                                  | None         | Write the 32-bit new USERCODE data to USERCODE register  |
| ISC_ERASE            | 00001110 | 0E  | 24 bits | D                  | —                                  | <sup>2</sup> | Bulk erase the memory array base on the access mode and array selection  |
| ISC_PROGRAM_DONE     | 01011110 | 5E  | 24 bits | D                  | —                                  | None         | Program the DONE bit if the device is in Configuration state.  |
| ISC_PROGRAM_SECURITY | 11001110 | CE  | 24 bits | C                  | —                                  | None         | Program the Security bit if the device is in Configuration state   |
| LSC_INIT_ADDRESS     | 01000110 | 46  | 24 bits | C                  | —                                  | None         | Initialize the Address Shift Register  |
| LSC_WRITE_ADDRESS    | 10110100 | B4  | 24 bits | B                  | —                                  | None         | Write the 16 bit Address Register to move the address quickly  |
| LSC_PROG_INCR        | 10000010 | 82  | 24 bits | B                  | 5,6                                | <sup>1</sup> | Write the configuration data to the configuration memory frame at current address and post increment the address.                |
| LSC_PROG_INCR_CMP    | 10111000 | B8  | 24 bits | B                  | —                                  | <sup>1</sup> | Decompress the configuration data; write the data to the configuration memory at current address and post increment the address. |
| LSC_READ_INCR        | 01101010 | 6A  | 24 bits | A                  | —                                  | None         | Read back the configuration memory data frames selected by the address register and post increment the address.                  |
| LSC_PROG_CTRL0       | 00100010 | 22  | 24 bits | B                  | —                                  | None         | Modify the Control Register 0  |
| LSC_READ_CTRL0       | 00100000 | 20  | 24 bits | A                  | —                                  | None         | Read the Control Register 0  |
| LSC_PROG_CTRL1       | 00100010 | 23  | 24 bits | B                  | —                                  | None         | Modify the Control Register 1  |
|                      |          |     |         |                    |                                    | <sup>1</sup> | Modify the NV content when access NVM  |
| LSC_READ_CTRL1       | 00100000 | 21  | 24 bits | A                  | —                                  | None         | Read the Control Register 1  |
| LSC_RESET_CRC        | 00111011 | 3B  | 24 bits | C                  | —                                  | None         | Reset 16-bit frame CRC register to 0x0000  |
| LSC_READ_CRC         | 01100000 | 60  | 24 bits | A                  | —                                  | None         | Read 16-bit frame CRC register content   |

| Command             | Binary   | Hex | Operand | Class <sup>4</sup> | Flow Operation Number <sup>5</sup> | Delay Time   | Description   |
|---------------------|----------|-----|---------|--------------------|------------------------------------|--------------|---|
| LSC_PROG_SED_CRC    | 10100010 | A2  | 24 bits | B                  | —                                  | None         | Program the calculated 32-bit CRC based on configuration bit values only into overall CRC register                    |
| LSC_READ_SED_CRC    | 10100100 | A4  | 24 bits | A                  | —                                  | None         | Read the 32-bit SED CRC   |
| LSC_PROG_PASSWORD   | 11110001 | F1  | 24 bits | B                  | —                                  | <sup>1</sup> | Program 128-bit password into the non-volatile memory (Efuse)   |
| LSC_READ_PASSWORD   | 11110010 | F2  | 24 bits | A                  | —                                  | None         | Read out the 128-bit password before activated for verification   |
| LSC_SHIFT_PASSWORD  | 10111100 | BC  | 24 bits | B                  | —                                  | None         | Shift in the 128 bits password to unlock for re-configuration (necessary when password protection feature is active). |
| LSC_PROG_CIPHER_KEY | 11110011 | F3  | 24 bits | B                  | —                                  | <sup>1</sup> | Program the 256-bit cipher key into the non-volatile memory   |
| LSC_READ_CIPHER_KEY | 11110100 | F4  | 24 bits | A                  | —                                  | None         | Read out the 256-bit cipher key before activated for verification   |
| LSC_PROG_FEATURE    | 11100100 | E4  | 24 bits | B                  | —                                  | <sup>1</sup> | Program security related feature such as Decrypt Enable, etc. base on the selection from the operand.                 |
| LSC_READ_FEATURE    | 11100111 | E7  | 24 bits | A                  | —                                  | None         | Read security related feature such as Decrypt Enable, etc. base on the selection from the operand.                    |
| LSC_PROG_FEABITS    | 11111000 | F8  | 24 bits | B                  | —                                  | <sup>1</sup> | Program security related feature bits.  |
| LSC_READ_FEABITS    | 11111011 | FB  | 24 bits | A                  | —                                  | None         | Read security related feature bits.   |
| LSC_PROG_OTP        | 11111001 | F9  | 24 bits | B                  | —                                  | <sup>1</sup> | Program security related One-Time-Programmable bits based on the selection from the operand.                          |
| LSC_READ_OTP        | 11111010 | FA  | 24 bits | A                  | —                                  | None         | Read security related One-Time-Programmable bits based on the selection from the operand.                             |
| LSC_WRITE_COMP_DIC  | 00000010 | 02  | 24 bits | B                  | —                                  | None         | Loads the most frequently used patterns into the device for decompressing compressed bitstreams.                      |
| LSC_INIT_BUS_ADDR   | 11110110 | F6  | 24 bits | B                  | —                                  | None         | Write the INIT Bus ID and Address Registers   |
| LSC_INIT_BUS_WRITE  | 01110010 | 72  | 24 bits | B                  | —                                  | None         | Write data through the INIT Bus   |
| LSC_INIT_BUS_READ   | 11110111 | F7  | 24 bits | A                  | —                                  | None         | Read back data through INIT Bus   |
| JUMP                | 01111110 | 7E  | 24 bits | B                  | -                                  | None         | Direct the Bitstream to a different SPI Boot PROM Address.  |
| LSC_PROG_SPI        | 00111010 | 3A  | 24 bits | B                  | —                                  | <sup>3</sup> | Connect SPI Host Port to the SPI interface to program the External SPI Flash, for JTAG-MSPI and SSPI-MSPI bridging    |

| Command                            | Binary   | Hex | Operand | Class <sup>4</sup> | Flow Operation Number <sup>5</sup> | Delay Time   | Description  |
|------------------------------------|----------|-----|---------|--------------------|------------------------------------|--------------|--|
| SSPI_PORT_CTRL                     | 11011001 | D9  | 24 bits | C                  | —                                  | None         | Support Slave SPI Dual/Quad mode or reset to Series mode as specified in OPERANDs  |
| LSC_IO_CONTROL                     | 01010100 | 54  | 24 bits | C                  | —                                  | None         | Control the release of the I/O Banks during configuration for early I/O release<br>Operand bit [1]=1; Release right I/O BANK (Bank 1 and Bank 2)<br>Operand bit [0]=1; Release left I/O BANK (Bank 6 and Bank 7) |
| LSC_AUTH_CTRL <sup>6, 7</sup>      | 01011000 | 58  | 24 bits | C                  | —                                  | None         | HMAC/ECDSA Authentication Control  |
| LSC_PROG_ECDSA_PUBKEY <sup>7</sup> | 01011001 | 59  | 24 bits | B                  | —                                  | <sup>1</sup> | Program the ECDSA Public Key   |
| LSC_READ_ECDSA_PUBKEY <sup>7</sup> | 01011010 | 5A  | 24 bits | B                  | —                                  | None         | Read Back the ECDSA Public Key   |
| LSC_PROG_SEC_BOOT                  | 01111111 | 7F  | 24 bits | B                  | —                                  |              | Secondary boot address override; 32 bits data to replace the hard coded secondary boot address.  |
| LSC_READ_DR_UES                    | 01011101 | 5D  | 24 bits | A                  | —                                  | None         | Read the dry-run User Electronic Signature shadow register.  |

**Notes:**

1. Delay time depends on data frame size and clock frequency. Duration could be in seconds. Use LSC\_CHECK\_BUSY to check the status.
2. Delay time depends on the FPGA SRAM Array size and clock frequency. Use LSC\_CHECK\_BUSY to check the status.
3. Delay time depends predominantly on the bitstream size and clock frequency. Duration could be in seconds.
4. The different command types are described in the [Command Waveforms](#) section.
5. The Flow Operation Number refer to the operation sequence in [Figure 6.9](#).
6. Bitstream Support Only.
7. For Certus-NX product family only.

## 6.10. JTAG Instruction Support

The ISC and programming instructions are executed on the first rising edge of the TCK after entering Run-Test/Idle State. When an instruction is nullified, the instruction is not executed and no registers are updated or captured by the instruction. However, the associated register is still connected to TDI and TDO on Shift-DR state. For more details, refer to IEEE 1149 and IEEE 1532 standards.

**Table 6.15. JTAG Instruction Table**

| JTAG Instruction | OPCODE   |     | Source Data<br>at<br>Capture-DR   | Target Data<br>at<br>Update-DR | Target Data<br>at<br>Runtest Idle | Shift<br>Register | Note          |
|------------------|----------|-----|-----------------------------------|--------------------------------|-----------------------------------|-------------------|---------------|
|                  | Binary   | Hex |                                   |                                |                                   |                   |               |
| EXTEST           | 00010101 | 15  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| INTEST           | 00101100 | 2C  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| CLAMP            | 01111000 | 78  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| HIGHZ            | 00011000 | 18  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| PRELOAD          | 00011100 | 1C  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| SAMPLE           | 00011100 | 1C  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| EXTEST_PULSE     | 00101101 | 2D  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| EXTEST_TRAIN     | 00101110 | 2E  | Refer to IEEE1149.1 Specification |                                |                                   |                   |               |
| ISC_NOOP         | 00110000 | 30  | N/A                               | N/A                            | N/A                               | N/A               | No Operation. |

| JTAG Instruction   | OPCODE   |     | Source Data at Capture-DR | Target Data at Update-DR | Target Data at Runtest Idle | Shift Register        | Note  |
|--------------------|----------|-----|---------------------------|--------------------------|-----------------------------|-----------------------|---|
|                    | Binary   | Hex |                           |                          |                             |                       |   |
| IDCODE             | 11100000 | E0  | IDCODE                    | N/A                      | N/A                         | 32-bit shift register | Read out the 32-bit hardware IDCODE of the device. If the USRID feature bit is set, this IDCODE becomes the user-defined IDCODE.  |
| UIDCODE            | 00011001 | 19  | IDCODE                    | N/A                      | N/A                         | 64-bit shift register | Reads the 64-bit UNIQUE ID, 56 bits set by Lattice, 8 bits set by customer.   |
| USERCODE           | 11000000 | C0  | USERCODE Register         | N/A                      | N/A                         | 32-bit shift register | Read the 32-bit USER Electronic Signature   |
| LSC_READ_STATUS    | 00111100 | 3C  | Status register           | N/A                      | N/A                         | 64-bit shift register | Read out the internal status such as busy, fail, and others. (64-bit)   |
| LSC_CHECK_BUSY     | 11110000 | F0  | BUSY FLAG                 | N/A                      | N/A                         | 1-bit Shift Register  | Read 1 bit busy flag to check the command execution status.   |
| LSC_DEVICE_CONTROL | 01111101 | 7D  | N/A                       | N/A                      | N/A                         | 8-bit shift register  | An 8-bit field in Operand 1 allows various device control, such as GSR, Sleep, Dry Run etc.<br>Bit 0: Cause a Global Set/Reset to occur on the device.<br>Bit 1: Launch a One-time check of SED.<br>Bit 2: Reserved<br>Bit 3: Cause a reset on configuration logic related FSM and flags.<br>Bit 4: Reserved for internal dry-run.<br>Bit [6:5]: Launch Dry-Run event.<br>2'b01 - goes to BOOT 0 only<br>2'b10 - goes to BOOT 2 only<br>2'b11 - goes to BOOT 1; if failed, goes to BOOT 2<br>2'b00 - no dry-run<br>Bit 7: CRC check bit |
| LSC_REFRESH        | 01111001 | 79  | N/A                       | N/A                      | N/A                         | Bypass                | Equivalent to toggle the PROGRAMN pin.  |

| JTAG Instruction     | OPCODE   |     | Source Data at Capture-DR | Target Data at Update-DR | Target Data at Runtest Idle | Shift Register        | Note   |
|----------------------|----------|-----|---------------------------|--------------------------|-----------------------------|-----------------------|--|
|                      | Binary   | Hex |                           |                          |                             |                       |  |
| ISC_ENABLE           | 11000110 | C6  | CONFIG_INFO               | ISC_CONFIG               | N/A                         | 8-bit shift register  | Enable the device for configuration. The device enters Access State. The I/Os are tristated with a weak pull down. Array selection and modal state setup effective after stay at RTI for more than 2 clock cycles.   |
| ISC_ENABLE_X         | 01110100 | 74  | CONFIG_INFO               | ISC_CONFIG               | N/A                         | 8-bit shift register  | Allow the transparent read in JTAG. The device enters Transparent mode and the I/O remain to be governed by user logic if the device is already being configured. Array selection and modal state setup effective after stay at RTI for more than two clock cycles.  |
| ISC_DISABLE          | 00100110 | 26  | N/A                       | N/A                      | N/A                         | Bypass                | If the device is in Access State, exit Access State and enters the Complete State. The wake-up sequence starts if the DONE bit is programmed. If the device is in Transparent Read State, exit the Transparent Read State and enter the Operational State or Unprogrammed State dependent on the DONE bit. |
| LSC_BITSTREAM_BURST  | 01111010 | 7A  | N/A                       | N/A                      | N/A                         | Bypass                | Program the device with the bitstream sent in through the JTAG port.   |
| ISC_PROGRAM_USERCODE | 11000010 | C2  | N/A                       | N/A                      | User Code Register          | 32-bit Shift Register | Write the 32-bit usercode into the device  |
| ISC_ERASE            | 00001110 | 0E  | N/A                       | N/A                      | ISC_SECTIR Register         | 16-bit Shift Register | Bulk erase the SRAM memory array base on the   |

| JTAG Instruction     | OPCODE   |     | Source Data<br>at<br>Capture-DR | Target Data<br>at<br>Update-DR | Target Data<br>at<br>Runtest Idle | Shift<br>Register      | Note  |
|----------------------|----------|-----|---------------------------------|--------------------------------|-----------------------------------|------------------------|---|
|                      | Binary   | Hex |                                 |                                |                                   |                        |   |
|                      |          |     |                                 |                                |                                   |                        | access mode and array selection   |
| ISC_PROGRAM_DONE     | 01011110 | 5E  | N/A                             | N/A                            | N/A                               | Bypass                 | Program the DONE bit if the device is in Configuration State.   |
| ISC_PROGRAM_SECURITY | 11001110 | CE  | N/A                             | N/A                            | N/A                               | Bypass                 | Program the Security Bit. if the device is in Configuration State   |
| LSC_INIT_ADDRESS     | 01000110 | 46  | N/A                             | N/A                            | N/A                               | Bypass                 | Initialize the Address Shift Register   |
| LSC_WRITE_ADDRESS    | 10110100 | B4  | N/A                             | 32-bit address register        | N/A                               | 32-bit shift Register  | Write the 16 Bit Address Register to move the address quickly. Note this is 32-bit but for CRAM address only the lower 16-bits are used.  |
| LSC_PROG_INCR        | 10000010 | 82  | N/A                             | N/A                            | Configuration Data Frame          | DSR                    | Write the configuration data to the configuration memory frame at current address and post increment the address. Byte 2 to byte 0 of the opcode indicates number of frames included in the operand field.                |
| LSC_PROG_INCR_CMP    | 10111000 | B8  | N/A                             | N/A                            | Configuration Data Frame          | 128-bit shift Register | Decompress the configuration data; write the data to the configuration memory at current address and post increment the address. Byte 2 to byte 0 of the opcode indicates number of frames included in the operand field. |
| LSC_READ_INCR        | 01101010 | 6A  | N/A                             | N/A                            | N/A                               | N/A                    | Read back the configuration memory data frames selected by the address register and post increment the address.   |
| LSC_PROG_CTRL0       | 00100010 | 22  | N/A                             | Control Register0              | N/A                               | 32-bit shift Register  | Modify the Control Register 0.  |

| JTAG Instruction    | OPCODE   |     | Source Data at Capture-DR | Target Data at Update-DR | Target Data at Runtest Idle | Shift Register         | Note  |
|---------------------|----------|-----|---------------------------|--------------------------|-----------------------------|------------------------|---|
|                     | Binary   | Hex |                           |                          |                             |                        |   |
| LSC_READ_CTRL0      | 00100000 | 20  | Control register0         | N/A                      | N/A                         | 32-bit shift Register  | Read the Control Register 0.  |
| LSC_PROG_CNTRL1     | 00100011 | 23  | N/A                       | Control Register1        | N/A                         | 32-bit shift Register  | Modify the Control Register 1.  |
| LSC_READ_CNTRL1     | 00100001 | 21  | Control register0         | N/A                      | N/A                         | 32-bit shift Register  | Read the Control Register 1.  |
| LSC_RESET_CRC       | 00111011 | 3B  | N/A                       | N/A                      | N/A                         | Bypass                 | Reset 16-bit CRC register to 0x0000.  |
| LSC_READ_CRC        | 01100000 | 60  | CRC Checksum              | N/A                      | N/A                         | 16-bit Shift Register  | Read 16-bit CRC register content.   |
| LSC_PROG_SED_CRC    | 10100010 | A2  | N/A                       | SED CRC                  | N/A                         | 32-bit shift Register  | Store the calculated 32-bit CRC based on the configuration bit values only.                           |
| LSC_READ_SED_CRC    | 10100100 | A4  | SED CRC                   | N/A                      | N/A                         | 32-bit shift Register  | Verify the 32-bit SED CRC   |
| LSC_PROG_PASSWORD   | 11110001 | F1  | N/A                       | Password                 | N/A                         | 128-bit shift Register | Program 128-bit password into the non-volatile memory (Internal EFUSE Memory)                         |
| LSC_READ_PASSWORD   | 11110010 | F2  | Password                  | N/A                      | N/A                         | 128-bit shift Register | Read out the 128-bit password before activated for verification                                       |
| LSC_SHIFT_PASSWORD  | 10111100 | BC  | N/A                       | Password Match Flag      | N/A                         | 128-bit shift Register | Shift in the password to unlock for re-configuration if the part is locked by the password.           |
| LSC_PROG_CIPHER_KEY | 11110011 | F3  | N/A                       | N/A                      | N/A                         | 256-bit shift register | Program the 256-bit AES key into the non-volatile memory, if 256-bit key is selected.                 |
| LSC_READ_CIPHER_KEY | 11110100 | F4  | N/A                       | N/A                      | N/A                         | 256-bit shift register | Read out the 256-bit AES key before activated for verification, if 256-bit key is selected.           |
| LSC_PROG_FEATURE    | 11100100 | E4  | N/A                       | N/A                      | N/A                         | 96-bit shift register  | Program security related feature such as Decrypt Enable, etc. base on the selection from the operand. |
| LSC_READ_FEATURE    | 11100111 | E7  | N/A                       | N/A                      | N/A                         | 96-bit shift register  | Read security related feature such as Decrypt Enable,   |



| JTAG Instruction   | OPCODE   |     | Source Data at Capture-DR | Target Data at Update-DR | Target Data at Runtest Idle | Shift Register         | Note  |
|--------------------|----------|-----|---------------------------|--------------------------|-----------------------------|------------------------|---|
|                    | Binary   | Hex |                           |                          |                             |                        |   |
|                    |          |     |                           |                          |                             |                        | and others. base on the selection from the operand.   |
| LSC_PROG_FEABITS   | 11111000 | F8  | N/A                       | N/A                      | N/A                         | 16-bit shift register  | Program security related feature bits.  |
| LSC_READ_FEABITS   | 11111011 | FB  | N/A                       | N/A                      | N/A                         | 16-bit shift register  | Read security related feature bits.   |
| LSC_PROG_OTP       | 11111001 | F9  | N/A                       | N/A                      | N/A                         | 32-bit shift register  | Program security related One-Time-Programmable bits based on the selection from the operand.  |
| LSC_READ_OTP       | 11111010 | FA  | N/A                       | N/A                      | N/A                         | 32-bit shift register  | Read security related One-Time-Programmable bits based on the selection from the operand.   |
| LSC_WRITE_COMP_DIC | 00000010 | 02  | N/A                       | Compression Dictionary   | N/A                         | 128-bit shift register | Loads the most frequently used patterns into the device for decompressing compressed bitstreams.  |
| LSC_INIT_BUS_ADDR  | 11110110 | F6  | N/A                       | Bus Address              | N/A                         | 32-bit Shift Register  | Write INIT Bus Address and ID Code Registers to move the address quickly  |
| LSC_INIT_BUS_WRITE | 01110010 | 72  | N/A                       | N/A                      | INIT Data Frame             | Frame Shift Register   | Write Data to INIT Bus,   |
| LSC_INIT_BUS_READ  | 11110111 | F7  | N/A                       | N/A                      | N/A                         | Frame Shift Register   | Readback Data from INIT Bus.  |
| LSC_PROG_SPI       | 00111010 | 3A  | N/A                       | N/A                      | N/A                         | SPI Flash PROM         | Connect the TDI, TDO, Shift-DR-N and TCK signals to BUSY, SPID, DI, and MCLK pin respectively. The length of the register is not defined for this particular instruction. |
| LSC_IO_CONTROL     | 01010100 | 54  | N/A                       | N/A                      | N/A                         | 8-bit shift register   | Control the release of the I/O Banks during configuration for early I/O release   |

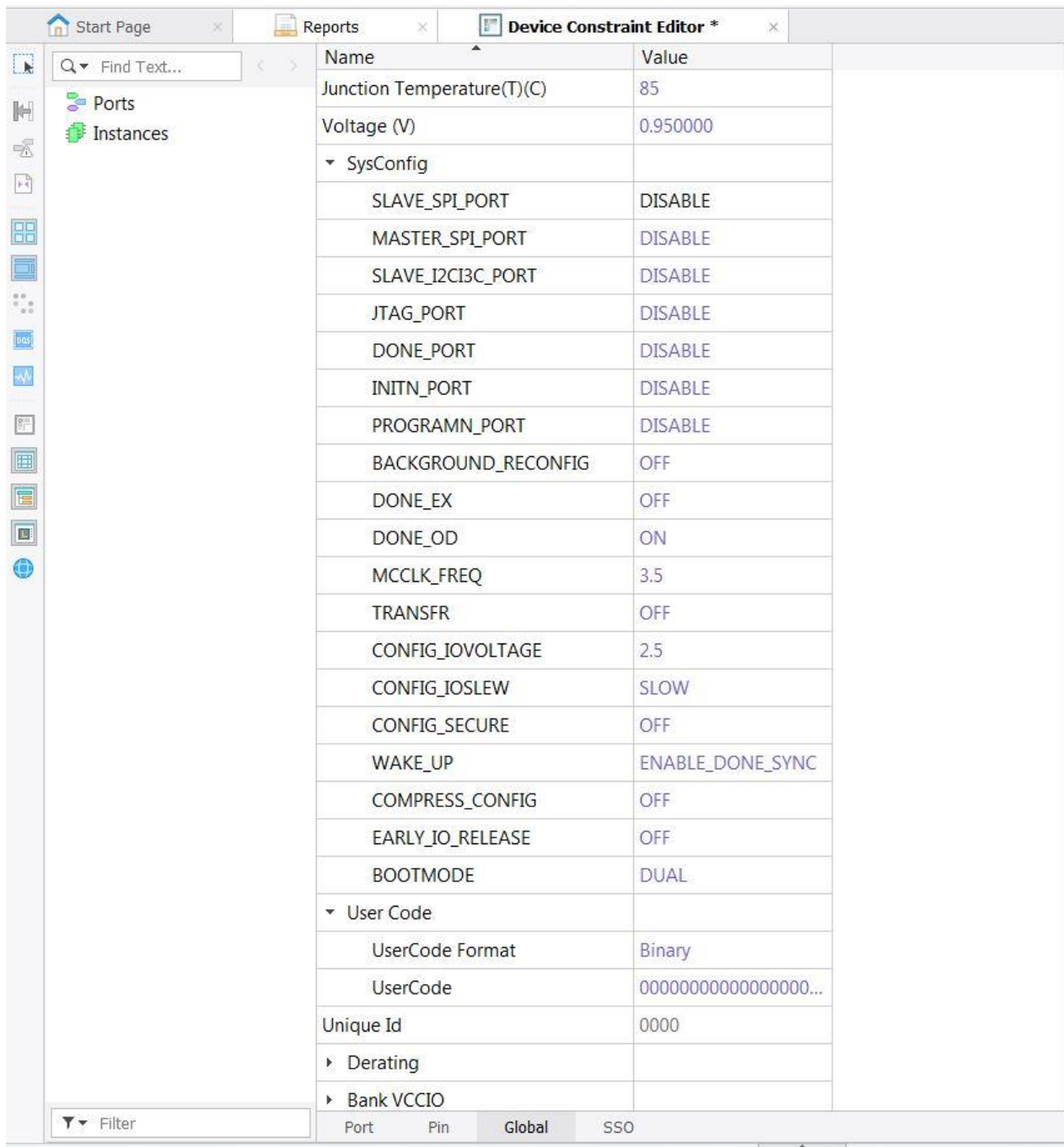
| JTAG Instruction       | OPCODE   |     | Source Data<br>at<br>Capture-DR | Target Data<br>at<br>Update-DR | Target Data<br>at<br>Runtest Idle | Shift<br>Register            | Note  |
|------------------------|----------|-----|---------------------------------|--------------------------------|-----------------------------------|------------------------------|---|
|                        | Binary   | Hex |                                 |                                |                                   |                              |   |
|                        |          |     |                                 |                                |                                   |                              | Bit [1]=1; Release<br>right I/O BANK<br>(Bank 1 and Bank 2)<br>Bit [0]=1; Release<br>left I/O BANK (Bank<br>6 and Bank 7) |
| LSC_PROG_ECDSA_PUBKEY1 | 01011001 | 59  | N/A                             | N/A                            | N/A                               | 512-bit<br>shift<br>register | Program the ECDSA<br>Public Key   |
| LSC_READ_ECDSA_PUBKEY1 | 01011010 | 5A  | N/A                             | N/A                            | N/A                               | 512-bit<br>shift<br>register | Read Back the<br>ECDSA Public Key   |
| LSC_READ_DR_UES        | 01011101 | 5D  | Dry Run UES                     | N/A                            | N/A                               | 32-bit<br>shift<br>Register  | Read the dry-run<br>User Electronic<br>Signature shadow<br>register.  |

**\*Note:** For Certus-NX product family only.

## 7. Software Selectable Options

The operation of the Nexus device configuration logic is managed by options selected in the Lattice Radiant design software. The Nexus device uses the built-in arbitration logic, as described in the [Configuration Ports Arbitration](#) section, to select the configuration port. User can set the configuration port persistence after device enters user function mode, and the system Configuration Pins (PROGRAMN pin, INITN pin, and DONE pin) persistence using the Lattice Radiant Device Constraint Editor.

The configuration logic preferences are accessed using Device Constraint Editor. Click on the Global tab, and look for the sysCONFIG tree. The sysCONFIG section is shown in [Figure 7.1](#).



| Name                       | Value               |
|----------------------------|---------------------|
| Junction Temperature(T)(C) | 85                  |
| Voltage (V)                | 0.950000            |
| ▼ SysConfig                |                     |
| SLAVE_SPI_PORT             | DISABLE             |
| MASTER_SPI_PORT            | DISABLE             |
| SLAVE_I2C3C_PORT           | DISABLE             |
| JTAG_PORT                  | DISABLE             |
| DONE_PORT                  | DISABLE             |
| INITN_PORT                 | DISABLE             |
| PROGRAMN_PORT              | DISABLE             |
| BACKGROUND_RECONFIG        | OFF                 |
| DONE_EX                    | OFF                 |
| DONE_OD                    | ON                  |
| MCCLK_FREQ                 | 3.5                 |
| TRANSFR                    | OFF                 |
| CONFIG_IOVOLTAGE           | 2.5                 |
| CONFIG_IOSLEW              | SLOW                |
| CONFIG_SECURE              | OFF                 |
| WAKE_UP                    | ENABLE_DONE_SYNC    |
| COMPRESS_CONFIG            | OFF                 |
| EARLY_IO_RELEASE           | OFF                 |
| BOOTMODE                   | DUAL                |
| ▼ User Code                |                     |
| UserCode Format            | Binary              |
| UserCode                   | 0000000000000000... |
| Unique Id                  | 0000                |
| ► Derating                 |                     |
| ► Bank VCCIO               |                     |

Filter

Port Pin Global SSO

**Figure 7.1. sysCONFIG Preferences in Global Tab, Lattice Radiant Device Constraint Editor**

**Table 7.1. sysCONFIG Options**

| Option Name  | Default Setting     | All Settings  |
|--|---------------------|---|
| SLAVE_SPI_PORT   | DISABLE             | DISABLE, SERIAL, DUAL, QUAD                             |
| MASTER_SPI_PORT  | DISABLE             | DISABLE, SERIAL, DUAL, QUAD                             |
| SLAVE_I2CI3C_PORT  | DISABLE             | DISABLE, ENABLE   |
| JTAG_PORT  | DISABLE             | DISABLE, ENABLE   |
| DONE_PORT  | DISABLE             | DISABLE, ENABLE   |
| INITN_PORT   | DISABLE             | DISABLE, ENABLE   |
| PROGRAMN_PORT  | DISABLE             | DISABLE, ENABLE   |
| BACKGROUND_RECONFIG  | OFF                 | OFF, ON, SRAM_EBR, SRAM_ONLY                            |
| DONE_EX  | OFF                 | OFF, ON   |
| DONE_OD  | ON                  | OFF, ON   |
| MCCLK_FREQ   | 3.5                 | 3.5, 7.0, 14.1, 28.1, 56.2, 90, 112.5                   |
| TRANSFR  | OFF                 | OFF, ON   |
| CONFIG_IOVOLTAGE   | 2.5                 | 2.5, 1.8, 3.3   |
| CONFIG_IOSLEW  | SLOW                | SLOW, MEDIUM, FAST                                      |
| CONFIG_SECURE  | OFF                 | OFF, ON   |
| WAKE_UP  | ENABLE_DONE_SYNC    | ENABLE_DONE_SYNC, DISABLE_DONE_SYNC                     |
| COMPRESS_CONFIG  | OFF                 | OFF, ON   |
| EARLY_IO_RELEASE   | OFF                 | OFF, ON   |
| BOOT_MODE  | DUAL                | DUAL, SINGLE, NONE                                      |
| USERCODE Format  | Binary              | Binary, Hex, ASCII, Auto                                |
| USERCODE   | 0 (32 binary zeros) | 32-bit  |
| UNIQUE_ID  | 0000                | Upper 16-bit user defined code for above Auto USERCODE. |
| <p><b>*Note:</b> The CONFIG_MODE option in Lattice Radiant Global Preference is for the software to preserve appropriated pins to help customer design flow. It does not serve the purpose of enabling SPI port nor setting appropriate bits in the PROM file.</p> |                     |   |

## 7.1. SLAVE\_SPI\_PORT

The SLAVE\_SPI\_PORT allows you to preserve the Slave SPI configuration port after the Nexus device enters user mode. There are four states to which the SLAVE\_SPI\_PORT preference can be set:

- **DISABLE** – This setting disconnects the SPI port pins from the configuration logic.
- **SERIAL** – This setting preserves the standard serial SPI port I/O (SCLK, SCSN, SI, SO) when the Nexus device is in user mode. When the pins are preserved, an external SPI master controller can interact with the configuration logic. The preference also prevents user from over-assigning I/O to those pins.
- **DUAL** – This setting preserves the SPI port I/O (SCLK, SCSN, SI/SD0, SO/SD1) in DUAL mode when the Nexus device is in user mode. When the pins are preserved, an external SPI master controller can interact with the configuration logic. The preference also prevents user from over-assigning I/O to those pins.
- **QUAD** – This setting preserves the SPI port I/O (SCLK, SCSN, SI/SD0, SO/SD1, SD2, SD3) in QUAD mode when the Nexus device is in user mode. When the pins are preserved, an external SPI master controller can interact with the configuration logic. The preference also prevents user from over-assigning I/O to those pins.

The SLAVE\_SPI\_PORT could be enabled at the same time as the MASTER\_SPI\_PORT, because the Slave SPI port and Master SPI port are in different I/O bank, this make the Nexus device possible to support the Slave SPI to Master SPI bridging functionality. The default setting for this preference is DISABLE.

## 7.2. MASTER\_SPI\_PORT

The MASTER\_SPI\_PORT allows you to preserve the Master SPI configuration port after the Nexus device enters user mode. There are four states to which the MASTER\_SPI\_PORT preference can be set:

- **DISABLE** – This setting disconnects the Master SPI port pins from the configuration logic. The Master SPI pins (MCLK, MCSN, MOSI/MD0, MISO/MD1, MD2 and MD3) could be general purpose user I/O.
- **SERIAL** – This setting preserves the SPI port I/O (MCLK, MCSN, MOSI/MD0, MISO/MD1) in serial mode when the Nexus device is in user mode. The preference also prevents user from over-assigning I/O to those pins.
- **DUAL** – This setting preserves the SPI port I/O (MCLK, MCSN, MOSI/MD0, MISO/MD1) in dual mode when the Nexus device is in user mode. The preference also prevents user from over-assigning I/O to those pins.
- **QUAD** – This setting preserves the SPI port I/O (MCLK, MCSN, MOSI/MD0, MISO/MD1, MD2 and MD3) in quad mode when the Nexus device is in user mode. The preference also prevents user from over-assigning I/O to those pins.

The default setting for this preference is DISABLE.

## 7.3. SLAVE\_I2CI3C\_PORT

The SLAVE\_I2CI3C\_PORT allows you to preserve the I<sup>2</sup>C/I3C configuration port after the Nexus device enters user mode. There are two states to which SLAVE\_I2CI3C\_PORT preference can be set:

- **ENABLE** – This setting preserve the I<sup>2</sup>C/I3C port pins (SD2/SCL, SD3/SDA) when the Nexus device is in user mode. It allows you to read all of the configuration data, except the EBR and the distributed RAM contents in user mode. The preference also prevents user from over-assigning I/O to those pins.
- **DISABLE** – This setting disconnects the I<sup>2</sup>C/I3C port pins (SD2/SCL, SD3/SDA) from the configuration logic. It allows I<sup>2</sup>C/I3C ports pins to be general purpose I/O, if they are not persisted for Slave SPI Quad mode as well.

The default setting for this preference is DISABLE.

## 7.4. JTAG\_PORT

The JTAG\_PORT allows you to preserve the JTAG configuration port after the Nexus device enters user mode. There are two states to which JTAG\_PORT preference can be set:

- **ENABLE** – This setting preserve the JTAG port pins (TCK, TMS, TDI and TDO) when the Nexus device is in user mode. The preference also prevents user from over-assigning I/O to those pins.
- **DISABLE** – This setting disconnects the JTAG port pins (TCK, TMS, TDI and TDO) from the configuration logic. It allows those ports pins to be general purpose I/O, if they are not persisted for Slave SPI port as well.

The default setting for this preference is DISABLE.

## 7.5. DONE\_PORT

The DONE\_PORT allows you to preserve the DONE pin as sysCONFIG pin for both configuration mode and user functional mode. There are two states to which DONE\_PORT preference can be set:

- **ENABLE** – This setting preserve the DONE pins as the sysCONFIG pin. This preference setting also prevents user from over-assigning I/O to this pin.
- **DISABLE** – This setting frees the DONE pin from sysCONFIG functionality. It allows this pin to be a general purpose I/O.

The default setting for this preference is DISABLE.

## 7.6. INITN\_PORT

The INIT\_PORT allows you to preserve the INITN pin as sysCONFIG pin for both configuration mode and user functional mode. There are two states to which INITN\_PORT preference can be set:

- **ENABLE** – This setting preserve the INITN pins as the sysCONFIG pin. This preference setting also prevents you from over-assigning I/O to this pin.
- **DISABLE** – This setting frees the INITN pin from sysCONFIG functionality. It allows this pin to be a general purpose I/O.

The default setting for this preference is DISABLE.

## 7.7. PROGRAMN\_PORT

The PROGRAMN\_PORT allows you to preserve the PROGRAMN pin as sysCONFIG pin for both configuration mode and user functional mode. There are two states to which PROGRAMN\_PORT preference can be set:

- **ENABLE** – This setting preserve the PROGRAMN pins as the sysCONFIG pin. This preference setting also prevents user from over-assigning I/O to this pin.
- **DISABLE** – This setting frees the PROGRAMN pin from sysCONFIG functionality. It allows this pin to be a general purpose I/O.

The default setting for this preference is DISABLE.

## 7.8. BACKGROUND\_RECONFIG

The BACKGROUND\_RECONFIG preference allows you to setup the device going to transparent access mode for the next PROGRAMN pin toggle or REFRESH command execution event. There are four states to which BACKGROUND\_RECONFIG preference can be set:

- **OFF** – This setting causes the device going to OFFLINE access mode for the next PROGRAMN pin toggle or REFRESH command execution event.
- **ON** – This setting causes the device going to transparent access mode with SRAM, EBR and IP Write capability for the next PROGRAMN pin toggle or REFRESH command execution event.
- **SRAM\_EBR** – This setting causes the device going to transparent access mode with SRAM and EBR Write capability for the next PROGRAMN pin toggle or REFRESH command execution event.
- **SRAM\_ONLY** – This setting causes the device going to transparent access mode with SRAM Write capability for the next PROGRAMN pin toggle or REFRESH command execution event.

The default setting for this preference is OFF.

## 7.9. DONE\_EX

You can select if the device should wake up on its own after the Done Bit is set or wait for an external DONE signal to drive the DONE Pin high. The DONE\_EX preference determines if the wake up sequence is driven by an external DONE signal. The DONE\_EX preference shall take a user entered ON or OFF. ON if you want to delay wake up until the DONE Pin is driven high by an external signal and synchronous to the clock. Select OFF if you want to synchronously wake up the device based on the internal DONE bit status and ignore any external driving of the DONE Pin. The default is OFF.

## 7.10. DONE\_OD

The DONE PIN is used in sysCONFIG to indicate that configuration is done. The DONE pin can be linked to other DONE pins and used to initiate the wake up process. The DONE pin can be open collector or active drive for the Nexus device. The default is ON. This insures your needs to make a conscious decision to drive the pin.

## 7.11. MCCLK Frequency

The MCLK\_FREQ preference allows you to alter the MCLK frequency used to retrieve data from an external SPI Flash when using EXTERNAL or Dual Boot configuration modes. The Nexus device uses a nominal 3.5 MHz ( $\pm 15\%$ ) clock frequency to begin retrieving data from the external SPI Flash. The MCLK\_FREQ value is stored in the incoming configuration data. The Nexus device reads a series of padding bits, a *start of data* word (0xBDB3) and a control register value. The control register contains the new MCLK\_FREQ value. The Nexus device switches to the new clock frequency shortly after receiving the MCLK\_FREQ value. The MCLK\_FREQ has a range of possible frequencies available from 3.5 MHz up to 112.5 MHz. Take care not to exceed the maximum clock rate of your SPI Flash, or of your printed circuit board.

## 7.12. TRANSFR

The TransFR function used by the Nexus device requires the configuration data loaded into the configuration SRAM, and any future configuration data file loaded into the external Flash memory have the TRANSFR set to the ENABLE state. See the TransFR Operation section and [TransFR Usage Guide for Nexus Platform \(FPGA-TN-02173\)](#) for more information about using TransFR.

## 7.13. CONFIG\_IOVOLTAGE

Because the sysCONFIG pins used for configuration may or may not be used in the design, you can declare the voltage interface for the sysCONFIG banks (bank 0 and bank 1). Setting this attribute informs the software which voltage is required in this bank to satisfy your sysCONFIG requirements. DRC errors can then be generated based on CONFIG\_IOVOLTAGE and usage of the dual-purpose sysCONFIG pins. The default is 2.5 V.

*CONFIG\_IOVOLTAGE = 1.8V/2.5V/3.3V (2.5V = Default)*

## 7.14. CONFIG\_IOSLEW

The CONFIG\_IOSLEW preference provides the option for different I/O slew rate for configuration related I/O, which makes the Nexus devices easily adaptable to different board environment. There are three options for the CONFIG\_IOSLEW preference, SLOW, MEDIUM and FAST. The default setting for the CONFIG\_IOSLEW preference is SLOW.

## 7.15. CONFIG\_SECURE

When the CONFIG\_SECURE preference set to ON, the read-back of the SRAM memory is blocked. The device must be reprogrammed in order to reset the security setting. Once the security fuses are reset, the device can be programmed again. The default setting for the CONFIG\_SECURE preference is OFF.

## 7.16. WAKE\_UP

The WAKE\_UP preference provides options for you to choose different wakeup sequence for releasing the global control signals when the device is entering the user functional mode.

- **ENABLE\_DONE\_SYNC** – Select the wakeup sequence synchronize with external DONE pin. For this option, the DONE\_EX preference has to be ON.
- **DISABLE\_DONE\_SYNC** – Select the wakeup sequence does not synchronize with external DONE pin. For this option, the DONE\_EX preference should be OFF.

The default selection for WAKE\_UP preference is set to ENABLE\_DONE\_SYNC.

See the [Device Wake-up Sequence](#) section for more detail about the Nexus device wake up sequence.

## 7.17. COMPRESS\_CONFIG

The COMPRESS\_CONFIG preference alters the way files are generated with compressed FPGA data frames. The COMPRESS\_CONFIG default setting is to be OFF.

## 7.18. EARLY\_IO\_RELEASE

The EARLY\_IO\_RELEASE preference enable early I/O release feature on I/O Bank1, I/O Bank2, I/O Bank6, and I/O Bank7. The EARLY\_IO\_RELEASE default setting is OFF.

## 7.19. BOOTMODE

The BOOTMODE preference allows you to select the booting mode at power up (POR), PROGRAMN pin toggle or REFRESH command execution.

- DUAL – Perform the dual boot for booting event. In case the primary booting image (bitstream) is failed, the secondary (Golden) booting image is automatically invoked to boot up the device.
- SINGLE – Perform the single boot only. If failed, device go to unprogrammed mode directly.
- NONE – No Master SPI booting at POR, PROGRAMN pin toggle or REFRESH command execution, wait for Slave Configuration Port to configure the device.

The default BOOTMODE selection is DUAL mode.

## 7.20. USERCODE\_FORMAT

The USERCODE\_FORMAT preference selects the format for the data field used to assign a value in the USERCODE preference. The USERCODE\_FORMAT has three options:

- Binary – USERCODE is set using 32 1 or 0 characters
- Hex – USERCODE is set using eight hexadecimal digits, that is 0-9A-F
- ASCII – USERCODE is set using up to four ASCII characters
- Auto – USERCODE is automatically created by the software. The upper 16 bits is Unique ID and the lower 16 bits are sequentially increased automatically for every bitstream generation.

## 7.21. USERCODE

The Nexus device contains a 32-bit register for storing a user-defined value. The default value stored in the register is 0x00000000. Using the USERCODE preference, you can assign any value to the register you desire. Suggested uses include the configuration data version number, a manufacturing ID code, date of assembly, or the JEDEC file checksum.

The format of the USERCODE field is controlled using the USERCODE\_FORMAT preference with Hex format as default. Data entry can be performed in either Binary, Hex, ASCII or Auto formats.

## 7.22. UNIQUE\_ID

The Nexus device contains a 16-bit register for storing a user-defined value. The default value stored in the register is 0x0000. Unique ID can only be set when USERCODE\_FORMAT is Auto.



## 8. Device Wake-up Sequence

When configuration is complete (the SRAM has been loaded), the device wakes up in a predictable fashion. If the Nexus device is the only device in the chain, or the last device in a chain, the wake-up process should be initiated by the completion of configuration. Once configuration is complete, the internal DONE bit is set and then the wake-up process begins. Figure 8.1 shows the wake-up sequence with DISABLE\_DONE\_SYNC option.

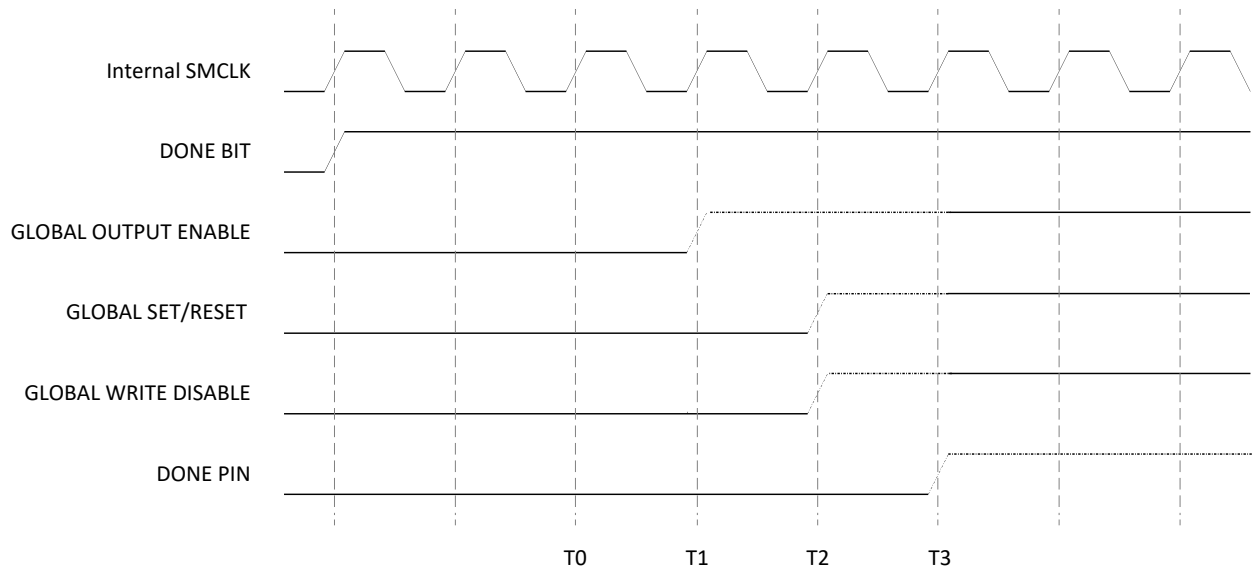


Figure 8.1. Wake-up Sequence Using Internal Clock

### 8.1. Wake-up Signals

Three internal signals, GSR, GWDIS, and GOE, determine the wake-up sequence.

- GSR is used to set and reset the core of the device. GSR is asserted (low) during configuration and de-asserted (high) in the wake-up sequence.
- When the GWDIS signal is low it safeguards the integrity of the RAM Blocks and LUTs in the device. This signal is low before the device wakes up. This control signal does not control the primary input pin to the device but controls specific control ports of EBR and LUTs.
- When low, GOE prevents the device's I/O buffers from driving the pins. The GOE only controls *output* pins. Once the internal DONE is asserted the Nexus device responds to input data.
- When high, the DONE pin indicates that configuration is complete and that no errors were detected.
- Before DONE pin goes high, all signals going to EBR should hold steady, otherwise, it might impact the EBR initialization.

The available wake-up sequences are shown in Table 8.1. A wake-up sequence is the order in which the signals change. The phase transition based on a wakeup clock, as discussed below. The exact timing relationship between the internal signals and the wakeup clock varies and is not specified.

Table 8.1. Wake-up Sequences

| Sequence          | Phase T0 | Phase T1 | Phase T2   | Phase T3 |
|-------------------|----------|----------|------------|----------|
| ENABLE_DONE_SYNC  | DONE     | GOE      | GWDIS, GSR | —        |
| DISABLE_DONE_SYNC | —        | GOE      | GWDIS, GSR | DONE     |

## 8.2. Wake-up Clock

The Nexus device always use the internal clock to perform the device wake up sequence. Once the Nexus device is configured, it enters the wake-up state, which is the transition between the configuration mode and user mode. This sequence is synchronized to the internal SMCLK, which is derived from internal oscillator.

## 9. Daisy Chaining

Typically, there is one configuration bitstream per FPGA in a system. Today's systems often have several FPGA devices. If all the FPGAs in the application utilize the same device and use the same bitstream, only a single bitstream is required. Using a ganged configuration loads multiple, similar FPGAs with the same bitstream at the same time.

However, to save PCB space and use external storage device more efficiently, several different FPGA bitstreams from various devices and designs can share a single configuration mechanism by using a daisy chain method.

The Nexus device supports two distinct daisy chaining methods. Bypass Option or Flow Through Option must be set for all Nexus devices in the sysCONFIG chain when using daisy chaining. The *Synchronous to External DONE pin* option (DONE\_EX) must be enabled in Lattice Radiant Device Constraint Editor. Based on the configuration ports arrangement, Nexus devices can only be the first device inside a configuration daisy chain.

### 9.1. Bypass Option

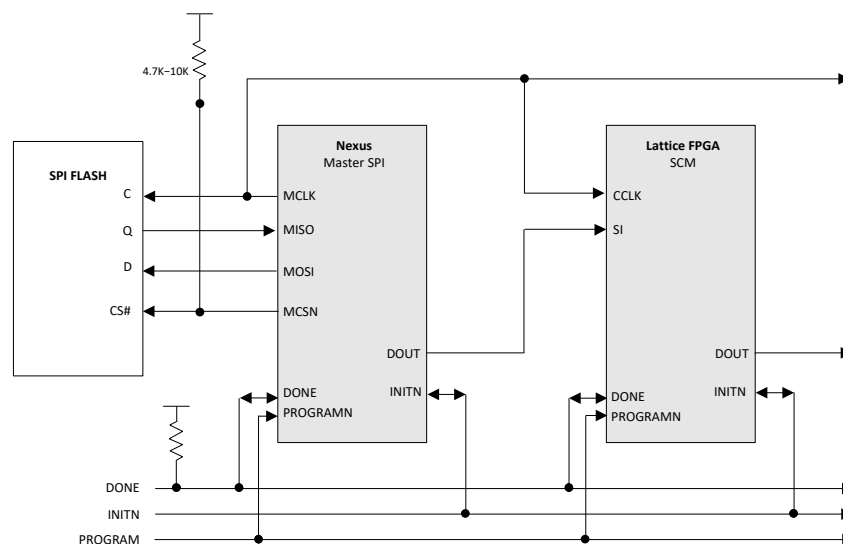
The Bypass option can be set by using the Diamond strategy properties, or a chain of bitstreams can be assembled and Bypass set using the Lattice Radiant Deployment Tool. The Bypass option is not supported when using dual-, or multi-boot configurations. The Bypass option is supported when using encrypted bitstream files with Nexus device.

When the first device completes configuration, and a Bypass command is included within the bitstream, any additional data coming into the FPGA configuration port overflows serially on DOUT. This data is applied to the DI pin of the next device (downstream devices must be set to Slave Serial mode).

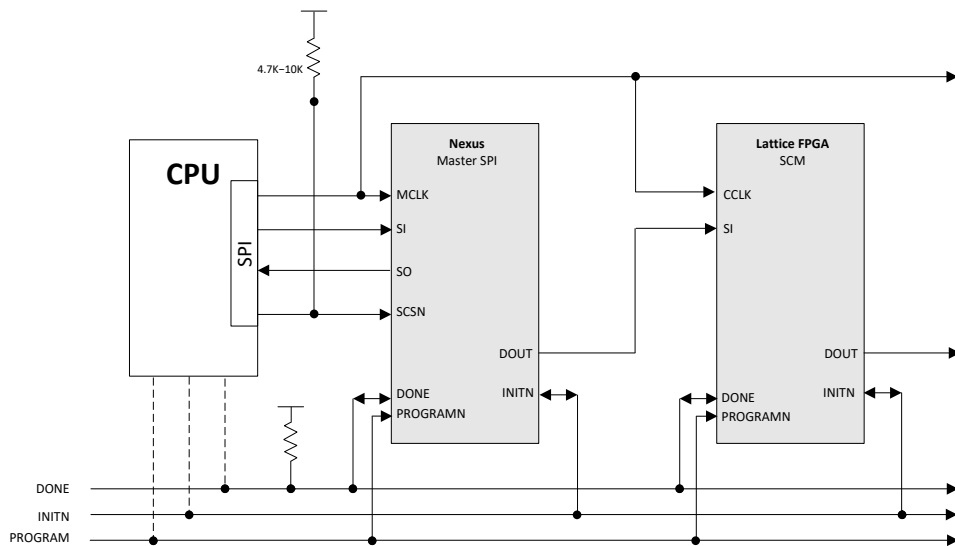
In Serial Configuration mode, the Bypass option connects DI to DOUT via a bypass register. The bypass register is initialized with a 1 at the beginning of configuration and stays at that value until the Bypass command is executed.

In parallel configuration modes, such as DUAL or QUAD mode, the Bypass option causes the excess data coming in on D[3:0] to be serially shifted to DOUT. The serialized data is shifted to DOUT through a bypass register. Once the Bypass option starts, the device remains in Bypass until the wake-up sequence completes.

Examples of the Nexus device in a configuration daisy chain with Bypass Option are shown in [Figure 9.1](#) and [Figure 9.2](#).



**Figure 9.1. Nexus in Configuration Daisy Chain with Master SPI in Bypass Mode**



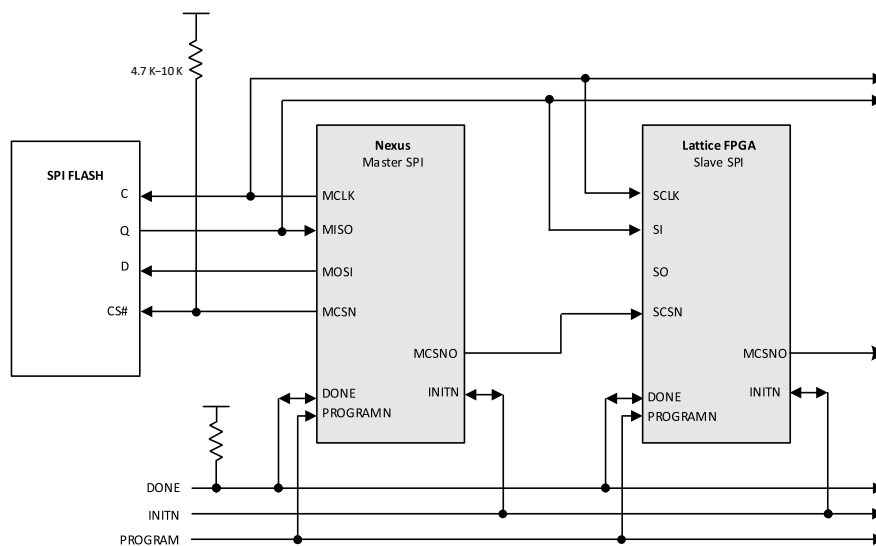
**Figure 9.2. Nexus in Configuration Daisy Chain with Slave SPI in Bypass Mode**

## 9.2. Flow Through Option

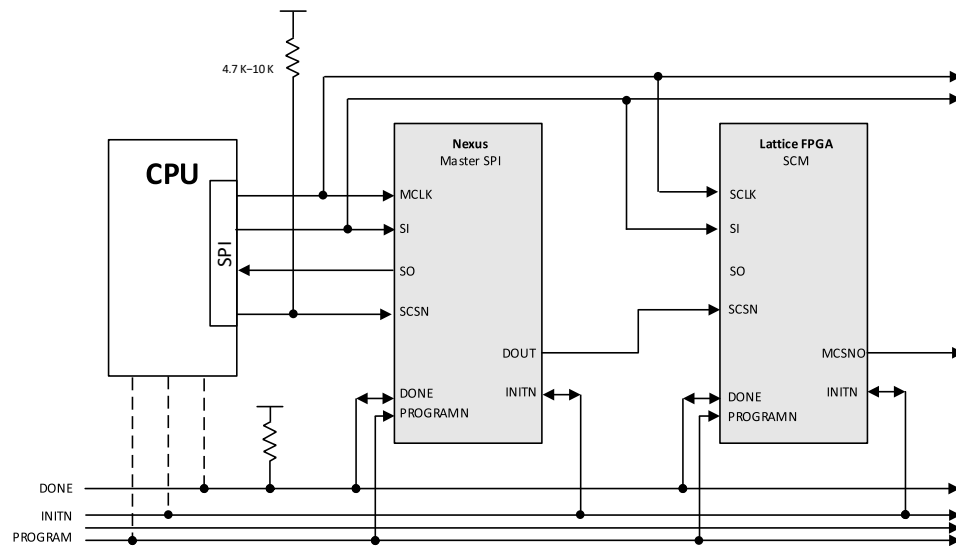
The Flow Through option for the Nexus device can only be used with serial daisy chains, and only support the Lattice FPGA with Slave SPI, which has the automatic bitstream mode.

When the first device completes configuration and a flow through command is included with the bitstream, the MCSNO pin is driven low. Once the flow through option starts, the device remains in flow through until the wake-up sequence completes.

Examples of the Nexus device in a configuration daisy chain with Flow Through option are shown in [Figure 9.3](#) and [Figure 9.4](#).



**Figure 9.3. Nexus in Configuration Daisy Chain with Master SPI in Flow Through Mode**



**Figure 9.4. Nexus in Configuration Daisy Chain with Slave SPI in Flow Through Mode**

## Appendix A. Nexus Slave SPI Programming Guide

The SPI port of the Nexus device can be used for device configuration. After configuration, the SPI pins can be used as user I/O except MCLK/SCLK, which is tristated. If these pins are not used by user logic, they are tristated with a weak pull-up. The Slave SPI port must be enabled in order to support device configuration from an external host or download cable using SPI protocol. This is done by setting the SLAVE\_SPI\_PORT preference to ENABLE in the bitstream via the Lattice Radiant Device Constraint Editor. Slave SPI mode supports single device configuration.

The Lattice Radiant Programmer supports the Slave SPI programming mode as one of the device access options. Selecting this option allows you to perform device erase, program, verify, readback, refresh, and more. The connections of Slave SPI pins to the Lattice programming cable are:

- TDI -> SISPI
- ISPEN -> SN
- TCK -> CCLK
- TDO -> SPISO

The Slave SPI chip select pin (SCSN) is held low during the command sequences. You can generate a SVF file from their bitstream (.bit) with the Lattice Deployment Tool software to show the details of the command sequences for Slave SPI programming mode.

The *Program, Erase and Verify* flow in Radiant Programmer for from any slave configuration port only target the FPGA SRAM array, which does not handle the Hard IP and EBR initialization. *Fast Program* option is preferred for Full device configuration.

## Appendix B. Nexus Bitstream File Format

### Configuration Bitstream Format

The base binary file format is the same for all non-encrypted, non-1532 configuration modes. Different file types (hex, binary, ASCII, and so on) may ultimately be used to configure the device, but the data in the file is the same. Table B.1 lists the format of a non-encrypted and compressed bitstream. The bitstream consists of a comment string, a header, the preamble, and the configuration setup and data.

Only the frame data can be compressed. The EBR data cannot be compressed. The data frame padding for compression is as follows:

- Before compression, pad the leftmost bits of the frame data with zeroes (0) to make the data frame 64-bit bounded. For example, if the original uncompressed data frame length is 125-bit, such as 01.....10, the data frame has to be padded with all 0 (3-bit 0s) at leftmost to make it 64-bit bounded (128-bit), 00001.....10.
- After compressing the frame data, pad the rightmost bits of the frame data with zeroes (0) to make the data frame byte bounded. For example, if the 128-bit data frame is compressed to become 101-bit (not byte bounded), such as 10.....01, the compressed data frame has to be padded with all 0 (3-bit 0s) at the rightmost to make it byte bounded (104-bit), 10.....01000

**Table B.1. Nexus Compressed Bitstream Format**

| Frame   | Contents[D7...D0]              | Description  | CRC           |
|---|--------------------------------|--|---------------|
| LSCC signature                                      | 4C534343                       | 32 bits of the LSCC Signature (LSCC = 0x4C534343)                                      | None          |
| Comments  | (Comment String)               | ASCII Comment (Argument) String and Terminator   |               |
| Header  | (LSB)<br>1111111111111111<br>1 | First 16 bits of the 32-bit Preamble   | None          |
|   | 101111011011001<br>1           | Second 16 bits of the 32-bit Preamble (0xFFFFBDB3)                                     |               |
|   | 11111111...111111<br>11        | 32-bit Dummy   |               |
| Frame (Reset CRC)                                   | 00111011                       | LSC_RESET_CRC Command  | Exclude       |
|   | 0                              | CRC Comparison Flag (0 = no)   | Exclude       |
|   | 0000.....000000                | 23-bit Command Information (23 zeroes)   | Exclude       |
| Dummy Frame   | 1111.....111111                | Dummy byte (32 bits of ones)   | Exclude       |
| Frame (Verify ID) OPTIONAL DATA But Mandatory Frame | 11100010                       | LSC_VERIFY_ID command (if the Verify ID data is not there, then Frame contains all 1s) | Start Include |
|   | 0                              | CRC Comparison Flag (0 = no) for Verify ID command                                     | Include       |
|   | 0000.....000000                | 23-bit Command Information (23 zeroes)   | Include       |
|   | (Device ID[31..0])             | 32 bit Device Id   | Include       |
| Frame (Control Register 0)                          | 00100010                       | LSC_PROG_CNTRL0 command  | Include       |
|   | 0                              | CRC Comparison Flag (0 = no)   | Include       |
|   | 0000.....000000                | 23-bit Command Information (23 zeroes).  | Include       |
|   | (CtlReg0[31..0])               | Control Register 0 Data definition <sup>3</sup>  | Include       |
| Frame (Control Register 1) OPTIONAL FRAME           | 00100011                       | LSC_PROG_CNTRL1 command  | Include       |
|   | 0                              | CRC Comparison Flag (0 = no)   | Include       |
|   | 0000.....000000                | 23-bit Command Information (23 zeroes).  | Include       |
|   | CtlReg1[31..0])                | Control Register 1 Data definition <sup>4</sup>  | Include       |

| Frame   | Contents[D7...D0]     | Description   | CRC               |
|---|-----------------------|---|-------------------|
| Frame<br>(Store<br>Compress) <sup>8</sup>             | 00000010              | LSC_WRITE_COMP_DIC Command.   | Include           |
|   | 0                     | CRC Comparison Flag (0 = no) for Verify ID command                                    | Include           |
|   | 0000.....000000       | 23-bit Command Information (23 zeroes).   | Include           |
|   | (Pattern16[7..0])     | Next most frequently occurring 8-bit pattern in the Frame Data.                       | Include           |
|   | Pattern15[7..0])      | Next most frequently occurring 8-bit pattern in the Frame Data.                       | Include           |
|   | =====                 | =====   | Include           |
|   | Pattern1[7..0])       | Next most frequently occurring 8-bit pattern in the Frame Data.                       | Include           |
| Frame<br>(Write Address<br>for Loading MIB<br>Frames) | 10110100              | LSC_WRITE_ADDRESS Command   | Include           |
|   | 0                     | CRC Comparison Flag (1 = yes).  | Include           |
|   | 0                     | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame) | Include           |
|   | 0                     | —   | Include           |
|   | 0                     | —   | Include           |
|   | 0000                  | —   | Include           |
|   | 0000.....000000       | 16-bit Command Information (23 zeroes).   | Include           |
|   | 000...00000           | 16 bit zeroes   | Include           |
|   | 1000000000000000<br>0 | 16 bit address to load  | Include           |
|   |                       |   |                   |
| Frame<br>(Auto Increment<br>For next 32<br>Frames)    | 10111000              | LSC_PROG_INCR_CMP Command.  | Include           |
|   | 1                     | CRC Comparison Flag (1 = yes) <sup>10</sup>   | Include           |
|   | 1                     | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame) | Include           |
|   | 0                     | Include dummy bits.   | Include           |
|   | 1                     | Dummy definition (1 = use Bit [3:0] as dummy byte count)                              | Include           |
|   | 0004                  | Dummy definition.   | Include           |
|   | 000000000010000<br>0  | Number of MIB Frames Always 32 (16-bit Frame Size)                                    | Include           |
| Frame<br>(Data 0)                                     | (Frame 0 Data)        | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>                          | Include           |
|   | 11111111...1111       | Dummy byte (32 bits of ones)  | Include           |
| Frame<br>(Data 1)                                     | (Frame 1 Data)        | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>                          | Include           |
|   | 11111111...1111       | Dummy byte (32 bits of ones)  | Include           |
|   | =====                 | =====   | =====             |
| Frame<br>(Data 31)                                    | (Frame 31 Data)       | x Data bits for Frame 31 (14 bits Parity + Data) <sup>6</sup>                         | Include: End      |
|   | (CRC[15..0])          | 16-bit CRC  | CRC               |
|   | 11111111...1111       | Dummy byte (32 bits of ones)  | Include:<br>Start |
| Frame<br>(Reset Address)                              | 01000110              | LSC_INIT_ADDRESS Command  | Include           |
|   | 0                     | CRC Comparison Flag (0 = no)  | Include           |
|   | 0000.....000000       | 23-bit Command Information (23 zeroes).   | Include           |



| Frame   | Contents[D7...D0]           | Description  | CRC               |
|---|-----------------------------|--|-------------------|
| Frame<br>(Write<br>Increment)                         | 10111000                    | LSC_PROG_INCR_CMP Command.   | Include           |
|   | 1                           | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include           |
|   | 1                           | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)  | Include           |
|   | 0                           | Include dummy bits.  | Include           |
|   | 1                           | Dummy definition (1 = use Bit [3:0] as dummy byte count)   | Include           |
|   | 0004                        | Dummy definition   | Include           |
|   | (16-bit = Number of Frames) | Number of configuration data frames included in operand <sup>6</sup><br>For Example, Jedi A<br>No. of Actual Data Frames = 4096(Total ASR Size) – 24(MIB Frames) – 12 (TAP Frames) | Include           |
| Frame (Data 0)  | (Frame 0 Data)              | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>   | Include           |
|   | 11111111...1111             | Dummy byte (32 bits of ones)   | Include           |
| Frame (Data 1)  | (Frame 1 Data)              | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>   | Include           |
|   | 11111111...1111             | Dummy byte (32 bits of ones)   | Include           |
| =====   | =====                       | =====  | =====             |
| Frame (Data n-1)                                      | (Frame n-1 Data)            | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC   | CRC               |
|   | 11111111...1111             | Dummy byte (32 bits of ones)   | Start:<br>Include |
| Frame<br>(Write address<br>for Loading TAP<br>Frames) | 10110100                    | LSC_WRITE_ADDRESS Command  | Include           |
|   | 0                           | CRC Comparison Flag (1 = yes).   | Include           |
|   | 0                           | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)  | Include           |
|   | 0                           | —  | Include           |
|   | 0                           | —  | Include           |
|   | 0000                        | —  | Include           |
|   | 0000.....000000             | 16-bit Command Information (23 zeroes).  | Include           |
|   | 000...00000                 | 16 bit Zeroes  | Include           |
|   | 100000000010000<br>0        | 16 bit address to load   | Include           |
| Frame<br>(Write<br>Increment )                        | 10111000                    | LSC_PROG_INCR_CMP Command.   | Include           |
|   | 1                           | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include           |
|   | 1                           | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)  | Include           |
|   | 0                           | Include dummy bits   | Include           |
|   | 1                           | Dummy definition (1 = use Bit [3:0] as dummy byte count)   | Include           |
|   | 0004                        | Dummy definition   | Include           |
|   | (16-bit = Frame Number)     | Number of TAP frames <sup>6</sup>  | Include           |
| Frame (Data 0)  | (Frame 0 Data)              | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>   | Include           |
|   | 11111111...1111             | Dummy byte (32 bits of ones)   | Include           |
| Frame (Data 1)  | (Frame 1 Data)              | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>   | Include           |
|   | 11111111...1111             | Dummy byte (32 bits of ones)   | Include           |
| =====   | =====                       | =====  | =====             |
| Frame (Data m-1)                                      | (Frame m-1 Data)            | x Data bits for Frame (m-1) (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC   | CRC               |
|   | 11111111...1111             | Dummy byte (32 bits of ones)   | Start:<br>Include |

| Frame   | Contents[D7...D0] | Description  | CRC                |
|---|-------------------|--|--------------------|
| Frame<br>(Power Control)                            | 01010110          | LSC_POWER_CTRL Command   | Include            |
|   | 0                 | CRC Comparison Flag (0 = no)   | Include            |
|   | 000....0000       | Operand 22 zeroes  | Include            |
|   | X                 | Enable the VCCPG Power Domain (1= yes)   | Include            |
| Frame (Init Bus<br>Address)                         | 11110110          | LSC_INIT_BUS_ADDR Command  | Include            |
|   | 0                 | CRC Comparison Flag (0 = no)   | Include            |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes)   | Include            |
|   | 00                | Default 2 zeroes (Reserved)  | Include            |
|   | Data[29:28]       | Bus Width <sup>5</sup><br>0: 8-bit Hard IP<br>1: 10-bit Hard IP<br>2: 10-bit EBR /Large RAM<br>3: 32-bit Hard IP   | Include            |
|   | Data [27:17]      | ID CODE  | Include            |
|   | Data[16:0]        | Starting address   | Include            |
| Frame<br>(Write data<br>through Init Bus)           | 01110010          | LSC_INIT_BUS_WRITE Command   | Include            |
|   | 1                 | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include            |
|   | 1                 | CRC Comparison at End Flag (1 = Execute CRC comparison at the end of all operands)   | Include            |
|   | 0                 | Exclude dummy bytes from bit stream  | Include            |
|   | 1                 | Dummy definition (1 = use the next 4-bit Dummy Definition settings)  | Include            |
|   | 0000              | Dummy definition   | Include            |
|   | (16-bit Size)     | Number of 72-bit frames in the operand. One = 0000000000000001   | Include            |
|   | (Data)            | Data Frame   | Include: End       |
|   | (CRC[15..0])      | 16-bit CRC   | CRC                |
| Frame<br>(SED CRC)<br>OPTIONAL<br>FRAME             | 10100010          | LSC_PROG_SED_CRC Command   | Start :<br>Include |
|   | 0                 | CRC Comparison Flag (0 = no)   | Include            |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes)   | Include            |
|   | (CRC[31..0])      | 32-bit SED CRC   | Include            |
| Frame<br>(Program<br>Security)<br>OPTIONAL<br>FRAME | 11001110          | ISC_PROGRAM_SECURITY Command   | Include            |
|   | 0                 | CRC Comparison Flag (0 = no)   | Include            |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes)   | Include            |
| Frame<br>(USERCODE)                                 | 11000010          | ISC_PROGRAM_USRCODE Command  | Include            |
|   | 1                 | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include            |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes)   | Include            |
|   | (U[31..0])        | 32-bit User code Data  | Include: End       |
|   | (CRC[15..0])      | 16-bit CRC   | CRC                |
| Frame<br>(Program Done)                             | 01011110          | ISC_PROGRAM_DONE   | Exclude            |
|   | 0                 | CRC Comparison Flag (0 = no)   | Exclude            |
|   | 00000....0000     | Operand 21 zeroes  | Exclude            |
|   | XX                | Behavior 00 = No overload (default), 01 = Serial System Configuration Daisy Chain Support in Bit stream, 10 = Parallel System Configuration Daisy Chain Support in bit stream, 11 = No overload(default) | Exclude            |
| Frame (DUMMY)                                       | 1111....1111      | 4 bytes DUMMY command (all ones).  | Exclude            |

**Notes:**

1. Data is byte bounded and must be padded with ones (1). For example, a 6-bit data frame of b111010 is written in the bitstream as b11111010 to make it byte bounded.
2. Only the data frame bits, highlighted in yellow, can be compressed.
3. If SED option is not chosen, place command DUMMY (NOOP) (32 bits of 1) on the command space. NOOP command is not included in CRC calculation. This is necessary to make the file size consistent with or without security option you chose. It must be replaced with the LSC\_PROG\_SED\_CRC frame by Bitgen if the you implement SED. Include in CRC calculation if LSC\_PROG\_SED\_CRC command. Do not include in CRC calculation if NOOP.
4. The CRC must be flipped so that CRC[0] is the LSB and CRC[15] is the MSB. For example, CRC = 0x0823 (CRC[15]...CRC[0]) in the bit file above is b1100010000010000. Refer to Appendix Generic Configuration Specification for CRC calculation details.
5. Insert a Write EBR Frame for each initialized EBR, or for each group of EBR Frames that is initialized to the same value. Omit if there is no EBR initialization data.
6. By default, this frame must be the Verify ID command. Bitgen shall have a switch to allow the Verify ID frame to be replaced with NOOP (all 1s). Include in CRC calculation if Verify ID command. Do not include in CRC calculation if NOOP.
7. If security option is not chosen, place command NOOP (32 bits of 1) on the command space. NOOP command is not included in CRC calculation. This is necessary to make the file size consistent with or without security option that you chose. Include in CRC calculation if Program Security command. Do not include in CRC calculation if NOOP.
8. The Control Register 0 must be set to the SW Default value, setting Control Register 0 bits [31..30] = [0..0].

**Table B.2. Nexus Uncompressed Bitstream Format – Without Early I/O Release**

| Frame   | Contents[D7...D0]         | Description  | CRC               |
|---|---------------------------|--|-------------------|
| LSCC signature  | 4C534343                  | 32 bits of the LSCC Signature (LSCC = 0x4C534343)  | None              |
| Comments  | (Comment String)          | ASCII Comment (Argument) String and Terminator   |                   |
| Header  | (LSB)<br>1111111111111111 | First 16 bits of the 32-bit Preamble   | None              |
|   | 101110110110011           | Second 16 bits of the 32-bit Preamble (0xFFFFBDB3)   |                   |
|   | 11111111...11111111       | 32-bit Dummy   |                   |
| Frame<br>(Reset CRC)  | 00111011                  | LSC_RESET_CRC Command  | Exclude           |
|   | 0                         | CRC Comparison Flag (0 = no)   | Exclude           |
|   | 0000.....000000           | 23-bit Command Information (23 zeroes)   | Exclude           |
| Dummy Frame   | 1111....111111            | Dummy byte (32 bits of ones)   | Exclude           |
| Frame<br>(Verify ID)<br>OPTIONAL DATA<br>But Mandatory<br>Frame | 11100010                  | LSC_VERIFY_ID command ( if the Verify ID data is not there,<br>then Frame contains all 1s) | Start Include     |
|   | 0                         | CRC Comparison Flag (0 = no) for Verify ID command   | Include           |
|   | 0000.....000000           | 23-bit Command Information (23 zeroes)   | Include           |
|   | (Device ID[31..0])        | 32 bit Device Id   | Include           |
| Frame<br>(Control<br>Register 0)                                | 00100010                  | LSC_PROG_CNTRL0 command  | Include           |
|   | 0                         | CRC Comparison Flag (0 = no)   | Include           |
|   | 0000.....000000           | 23-bit Command Information (23 zeroes)   | Include           |
|   | (CtlReg0[31..0])          | Control Register 0 Data definition <sup>3</sup>  | Include           |
| Frame<br>(Control<br>Register 1)<br>OPTIONAL<br>FRAME           | 00100011                  | LSC_PROG_CNTRL1 command  | Include           |
|   | 0                         | CRC Comparison Flag (0 = no)   | Include           |
|   | 0000.....000000           | 23-bit Command Information (23 zeroes)   | Include           |
|   | CtlReg1[31..0])           | Control Register 1 Data definition <sup>4</sup>  | Include           |
| Frame<br>(Write Address<br>for Loading MIB<br>Frames)           | 10110100                  | LSC_WRITE_ADDRESS Command  | Include           |
|   | 0                         | CRC Comparison Flag (1 = yes)  | Include           |
|   | 0                         | CRC Comparison at End Flag (0 = Execute CRC comparison at the<br>end of each data frame)   | Include           |
|   | 0                         | —  | Include           |
|   | 0                         | —  | Include           |
|   | 0000                      | —  | Include           |
|   | 0000.....000000           | 16-bit Command Information (23 zeroes).  | Include           |
|   | 000...00000               | 16 bit zeroes  | Include           |
|   | 1000000000000000          | 16 bit address to load <sup>7</sup>  | Include           |
| Frame<br>(Auto Increment<br>For next 32<br>Frames)              | 10000010                  | LSC_PROG_INCR command  | Include           |
|   | 1                         | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include           |
|   | 0                         | CRC Comparison at End Flag (0 = Execute CRC comparison at the<br>end of each data frame)   | Include           |
|   | 0                         | Include dummy bits   | Include           |
|   | 1                         | Dummy definition (1 = use Bit [3:0] as dummy byte count)                                   | Include           |
|   | 0001                      | Dummy definition.  | Include           |
|   | 00000000000100000         | (MIB Frames size 32) (16-bit Frame Size) <sup>6</sup>                                      | Include           |
| Frame<br>(Data 0)   | (Frame 0 Data)            | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>                               | Include: End      |
|   | (CRC[15..0])              | 16-bit CRC   | CRC               |
|   | 11111111                  | Dummy byte (8 bits of ones)  | Start:<br>Include |
| Frame   | (Frame 1 Data)            | x Data bits for Frame 1(14 bits Parity + Data) <sup>6</sup>                                | Include: End      |

| Frame   | Contents[D7...D0]           | Description   | CRC               |
|---|-----------------------------|---|-------------------|
| (Data 1)  | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| =====   | =====                       | =====   | =====             |
| Frame<br>(Data 31)                                    | (Frame 31 Data)             | x Data bits for Frame 31 (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame<br>(Reset Address)                              | 01000110                    | LSC_INIT_ADDRESS Command  | Include           |
|   | 0                           | CRC Comparison Flag (0 = no)  | Include           |
|   | 0000.....000000             | 23-bit Command Information (23 zeroes)  | Include           |
| Frame<br>(Write<br>Increment)                         | 10000010                    | LSC_PROG_INCR command   | Include           |
|   | 1                           | CRC Comparison Flag (1 = yes) <sup>10</sup>   | Include           |
|   | 0                           | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)   | Include           |
|   | 0                           | Include dummy bits  | Include           |
|   | 1                           | Dummy definition (1 = use Bit [3:0] as dummy byte count)  | Include           |
|   | 0001                        | Dummy definition.   | Include           |
|   | (16-bit = Number of Frames) | Number of configuration data frames included in operand <sup>6</sup><br>For Example Jedi A<br>No. of Actual Data Frames = 4072(Total ASR Size) – 32(MIB Frames) – 12 (TAP Frames) | Include           |
| Frame (Data 0)  | (Frame 0 Data)              | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame (Data 1)  | (Frame 1 Data)              | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| =====   | =====                       | =====   | =====             |
| Frame (Data n-1)                                      | (Frame n-1 Data)            | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame<br>(Write address<br>for Loading TAP<br>Frames) | 10110100                    | LSC_WRITE_ADDRESS Command   | Include           |
|   | 0                           | CRC Comparison Flag (1 = yes)   | Include           |
|   | 0                           | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)   | Include           |
|   | 0                           | —   | Include           |
|   | 0                           | —   | Include           |
|   | 0000                        | —   | Include           |
|   | 0000.....000000             | 16-bit Command Information (23 zeroes)  | Include           |
|   | 000...00000                 | 16 bit Zeroes   | Include           |
|   | 1000000000100000            | 16 bit address to load <sup>7</sup>   | Include           |

| Frame                                  | Contents[D7...D0]       | Description  | CRC            |
|--|-------------------------|--|----------------|
| Frame<br>(Write Increment)             | 10000010                | LSC_PROG_INCR command  | Include        |
|  | 1                       | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include        |
|  | 0                       | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)                            | Include        |
|  | 0                       | Include dummy bits   | Include        |
|  | 1                       | Dummy definition (1 = use Bit [3:0] as dummy byte count)   | Include        |
|  | 0001                    | Dummy definition   | Include        |
|  | (16-bit = Frame Number) | Number of TAP frames <sup>6</sup>  | Include        |
| Frame (Data 0)                         | (Frame 0 Data)          | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>   | Include: End   |
|  | (CRC[15..0])            | 16-bit CRC   | CRC            |
|  | 11111111                | Dummy byte (8 bits of ones)  | Start: Include |
| Frame (Data 1)                         | (Frame 1 Data)          | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>   | Include: End   |
|  | (CRC[15..0])            | 16-bit CRC   | CRC            |
|  | 11111111                | Dummy byte (8 bits of ones)  | Start: Include |
| =====                                  | =====                   | =====  | =====          |
| Frame (Data m-1)                       | (Frame m-1 Data)        | x Data bits for Frame (m-1) (14 bits Parity + Data) <sup>6</sup>   | Include: End   |
|  | (CRC[15..0])            | 16-bit CRC   | CRC            |
|  | 11111111                | Dummy byte (8 bits of ones)  | Start: Include |
| Frame<br>(Power Control)               | 01010110                | LSC_POWER_CTRL Command   | Include        |
|  | 0                       | CRC Comparison Flag (0 = no)   | Include        |
|  | 000....0000             | Operand 22 zeroes  | Include        |
|  | X                       | Enable the VCCPG Power Domain (1= yes)   | Include        |
| Frame<br>(Init Bus Address)            | 11110110                | LSC_INIT_BUS_ADDR Command  | Include        |
|  | 0                       | CRC Comparison Flag (0 = no)   | Include        |
|  | 0000.....000000         | 23-bit Command Information (23 zeroes).  | Include        |
|  | 00                      | Default 2 zeroes (Reserved)  | Include        |
|  | Data[29:28]             | Bus Width <sup>5</sup><br>0: 8-bit Hard IP<br>1: 10-bit Hard IP<br>2: 10-bit EBR /Large RAM<br>3: 32-bit Hard IP | Include        |
|  | Data [27:17]            | ID CODE  | Include        |
|  | Data[16:0]              | Starting address   | Include        |
| Frame<br>(Write data through Init Bus) | 01110010                | LSC_INIT_BUS_WRITE Command   | Include        |
|  | 1                       | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include        |
|  | 1                       | CRC Comparison at End Flag (1 = Execute CRC comparison at the end of all operands)                               | Include        |
|  | 0                       | Exclude dummy bytes from bit stream.   | Include        |
|  | 1                       | Dummy definition (1 = use the next 4-bit Dummy Definition settings)  | Include        |
|  | 0000                    | Dummy definition.  | Include        |
|  | (16-bit Size)           | Number of 72-bit frames in the operand. One = 0000000000000001   | Include        |
|  | (Data)                  | Data Frame   | Include: End   |
|  | (CRC[15..0])            | 16-bit CRC   | CRC            |

| Frame   | Contents[D7...D0] | Description   | CRC                |
|---|-------------------|---|--------------------|
| Frame<br>(SED CRC)<br>OPTIONAL<br>FRAME             | 10100010          | LSC_PROG_SED_CRC Command  | Start :<br>Include |
|   | 0                 | CRC Comparison Flag (0 = no)  | Include            |
|   | 0000.....0000000  | 23-bit Command Information (23 zeroes)  | Include            |
|   | (CRC[31..0])      | 32-bit SED CRC  | Include            |
| Frame<br>(Program<br>Security)<br>OPTIONAL<br>FRAME | 11001110          | ISC_PROGRAM_SECURITY Command  | Include            |
|   | 0                 | CRC Comparison Flag (0 = no)  | Include            |
|   | 0000.....0000000  | 23-bit Command Information (23 zeroes)  | Include            |
| Frame<br>(USERCODE)                                 | 11000010          | ISC_PROGRAM_USRCODE Command   | Include            |
|   | 1                 | CRC Comparison Flag (1 = yes) <sup>10</sup>   | Include            |
|   | 0000.....0000000  | 23-bit Command Information (23 zeroes)  | Include            |
|   | (U[31..0])        | 32-bit User code Data   | Include: End       |
|   | (CRC[15..0])      | 16-bit CRC  | CRC                |
| Frame<br>(Program Done)                             | 01011110          | ISC_PROGRAM_DONE  | Exclude            |
|   | 0                 | CRC Comparison Flag (0 = no)  | Exclude            |
|   | 00000....0000     | Operand 21 zeroes   | Exclude            |
|   | XX                | Behavior 00 = No overload(default) , 01 = Serial System Configuration Daisy Chain Support in Bit stream , 10 = Parallel System Configuration Daisy Chain Support in bit stream, 11 = No overload(default) | Exclude            |
| Frame (DUMMY)                                       | 1111...1111       | 4 bytes DUMMY command (all ones)  | Exclude            |

**Notes:**

1. Data is byte bounded and must be padded with ones (1). For example, a 6-bit data frame of b111010 is written in the bitstream as b11111010 to make it byte bounded.
2. Only the data frame bits, highlighted in yellow, can be compressed.
3. If SED option is not chosen, place command DUMMY (NOOP) (32 bits of 1) on the command space. NOOP command is not included in CRC calculation. This is necessary to make the file size consistent with or without security option you chose. It must be replaced with the LSC\_PROG\_SED\_CRC frame by Bitgen if you implement SED. Include in CRC calculation if LSC\_PROG\_SED\_CRC command. Do not include in CRC calculation if NOOP.
4. The CRC must be flipped so that CRC[0] is the LSB and CRC[15] is the MSB. For example, CRC = 0x0823 (CRC[15]...CRC[0]) in the bit file above is b1100010000010000. Refer to Appendix Generic Configuration Specification for CRC calculation details.
5. Insert a Write EBR Frame for each initialized EBR, or for each group of EBR Frames that is initialized to the same value. Omit if there is no EBR initialization data.
6. By default, this frame must be the Verify ID command. Bitgen shall have a switch to allow the Verify ID frame to be replaced with NOOP (all 1s). Include in CRC calculation if Verify ID command. Do not include in CRC calculation if NOOP.
7. If security option is not chosen, place command NOOP (32 bits of 1) on the command space. NOOP command is not included in CRC calculation. This is necessary to make the file size consistent with or without security option that you chose. Include in CRC calculation if Program Security command. Do not include in CRC calculation if NOOP.
8. The Control Register 0 must be set to the SW Default value, setting Control Register 0 bits [31..30] = [0..0].

**Table B.3. Nexus Uncompressed Bitstream Format – With Early I/O Release**

| Frame  | Contents[D7...D0]              | Description   | CRC               |
|--|--------------------------------|---|-------------------|
| LSCC signature   | 4C534343                       | 32 bits of the LSCC Signature (LSCC = 0x4C534343)                                       | None              |
| Comments   | (Comment String)               | ASCII Comment (Argument) String and Terminator  |                   |
| Header   | (LSB)<br>1111111111111111<br>1 | First 16 bits of the 32-bit Preamble.   | None              |
|  | 10111011011001<br>1            | Second 16 bits of the 32-bit Preamble (0xFFFFBDB3).                                     |                   |
|  | 11111111...11111<br>111        | 32-bit Dummy  |                   |
| Frame (Reset CRC)  | 00111011                       | LSC_RESET_CRC Command   | Exclude           |
|  | 0                              | CRC Comparison Flag (0 = no).   | Exclude           |
|  | 0000.....000000                | 23-bit Command Information (23 zeroes).   | Exclude           |
| Dummy Frame  | 1111....111111                 | Dummy byte (32 bits of ones)  | Exclude           |
| Frame (Verify ID)<br>OPTIONAL DATA<br>But Mandatory<br>Frame | 11100010                       | LSC_VERIFY_ID command ( if the Verify ID data is not there, then Frame contains all 1s) | Start:<br>Include |
|  | 0                              | CRC Comparison Flag (0 = no) for Verify ID command                                      | Include           |
|  | 0000.....000000                | 23-bit Command Information (23 zeroes).   | Include           |
|  | (Device ID[31..0])             | 32 bit Device Id  | Include           |
| Frame (Control<br>Register 0)                                | 00100010                       | LSC_PROG_CNTRL0 command   | Include           |
|  | 0                              | CRC Comparison Flag (0 = no)  | Include           |
|  | 0000.....000000                | 23-bit Command Information (23 zeroes).   | Include           |
|  | (CtlReg0[31..0])               | Control Register 0 Data definition <sup>3</sup>   | Include           |
| Frame (Control<br>Register 1)<br>OPTIONAL FRAME              | 00100011                       | LSC_PROG_CNTRL1 command   | Include           |
|  | 0                              | CRC Comparison Flag (0 = no)  | Include           |
|  | 0000.....000000                | 23-bit Command Information (23 zeroes).   | Include           |
|  | CtlReg1[31..0])                | Control Register 1 Data definition <sup>4</sup>   | Include           |
| Frame<br>(Write Address for<br>Loading MIB<br>Frames)        | 10110100                       | LSC_WRITE_ADDRESS Command   | Include           |
|  | 0                              | CRC Comparison Flag (1 = yes).  | Include           |
|  | 0                              | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)   | Include           |
|  | 0                              | —   | Include           |
|  | 0                              | —   | Include           |
|  | 0000                           | —   | Include           |
|  | 0000.....000000                | 16-bit Command Information (23 zeroes).   | Include           |
|  | 000...00000                    | 16 bit zeroes   | Include           |
|  | 1000000000000000               | 16 bit address to load <sup>7</sup>   | Include           |
| Frame<br>(Auto Increment<br>For next 32 Frames)              | 10000010                       | LSC_PROG_INCR command   | Include           |
|  | 1                              | CRC Comparison Flag (1 = yes) <sup>10</sup>   | Include           |
|  | 0                              | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)   | Include           |
|  | 0                              | Include dummy bits.   | Include           |
|  | 1                              | Dummy definition (1 = use Bit [3:0] as dummy byte count)                                | Include           |
|  | 0001                           | Dummy definition.   | Include           |
|  | 0000000000001000<br>00         | Number of MIB Frames Always 32 (16-bit Frame Size <sup>6,7</sup> )                      | Include           |



| Frame   | Contents[D7...D0]           | Description   | CRC               |
|---|-----------------------------|---|-------------------|
| Frame<br>(Data 0)   | (Frame 0 Data)              | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame<br>(Data 1)   | (Frame 1 Data)              | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| =====   | =====                       | =====   | =====             |
| Frame<br>(Data 31)  | (Frame 31 Data)             | x Data bits for Frame 31 (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame<br>(Release I/O)<br>OPTIONAL FRAME<br>(BOTH BANK A and B) | 01010100                    | LSC_IO_CONTROL Command  | Include           |
|   | 0                           | CRC Comparison Flag (0 = no)  | Include           |
|   | 00000                       | 5 zeroes  | Include           |
|   | X                           | release I/O bank B  | Include           |
|   | X                           | release I/O bank A  | Include           |
|   | 000....0000                 | all 16 zeroes in Operand  |                   |
| Dummy Frame   | 0xFFFFF...FFFF              | 1000 Bytes dummy  | Include           |
| Frame<br>(Reset Address)  | 01000110                    | LSC_INIT_ADDRESS Command  | Include           |
|   | 0                           | CRC Comparison Flag (0 = no)  | Include           |
|   | 0000.....000000             | 23-bit Command Information (23 zeroes).   | Include           |
| Frame<br>(Write Increment)                                      | 10000010                    | LSC_PROG_INCR command   | Include           |
|   | 1                           | CRC Comparison Flag (1 = yes) <sup>10</sup>   | Include           |
|   | 0                           | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)   | Include           |
|   | 0                           | Include dummy bits.   | Include           |
|   | 1                           | Dummy definition (1 = use Bit [3:0] as dummy byte count)  | Include           |
|   | 0001                        | Dummy definition.   | Include           |
|   | (16-bit = Number of Frames) | Number of configuration data frames included in operand <sup>6</sup><br>For Example Jedi A<br>No. of Actual Data Frames = 4096(Total ASR Size) – 24(MIB Frames) – 12 (TAP Frames) | Include           |
| Frame (Data 0)  | (Frame 0 Data)              | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame (Data 1)  | (Frame 1 Data)              | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| =====   | =====                       | =====   | =====             |
| Frame (Data n-1)  | (Frame n-1 Data)            | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>  | Include: End      |
|   | (CRC[15..0])                | 16-bit CRC  | CRC               |
|   | 11111111                    | Dummy byte (8 bits of ones)   | Start:<br>Include |
| Frame   | 10110100                    | LSC_WRITE_ADDRESS Command   | Include           |
|   | 0                           | CRC Comparison Flag (1 = yes).  | Include           |

| Frame                                  | Contents[D7...D0]       | Description  | CRC               |
|--|-------------------------|--|-------------------|
| (Write address for Loading TAP Frames) | 0                       | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)                            | Include           |
|  | 0                       | —  | Include           |
|  | 0                       | —  | Include           |
|  | 0000                    | —  | Include           |
|  | 0000.....000000         | 16-bit Command Information (23 zeroes).  | Include           |
|  | 000...00000             | 16 bit Zeroes  | Include           |
|  | 100000000010000<br>0    | 16 bit address to load   | Include           |
| Frame<br>(Write Increment )            | 10000010                | LSC_PROG_INCR command  | Include           |
|  | 1                       | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include           |
|  | 0                       | CRC Comparison at End Flag (0 = Execute CRC comparison at the end of each data frame)                            | Include           |
|  | 0                       | Include dummy bits.  | Include           |
|  | 1                       | Dummy definition (1 = use Bit [3:0] as dummy byte count)   | Include           |
|  | 0001                    | Dummy definition.  | Include           |
|  | (16-bit = Frame Number) | Number of TAP frames <sup>6</sup>  | Include           |
| Frame (Data 0)                         | (Frame 0 Data)          | x Data bits for Frame 0 (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|  | (CRC[15..0])            | 16-bit CRC   | CRC               |
|  | 11111111                | Dummy byte (8 bits of ones)  | Start:<br>Include |
| Frame (Data 1)                         | (Frame 1 Data)          | x Data bits for Frame 1 (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|  | (CRC[15..0])            | 16-bit CRC   | CRC               |
|  | 11111111                | Dummy byte (8 bits of ones)  | Start:<br>Include |
| =====                                  | =====                   | =====  | =====             |
| Frame (Data m-1)                       | (Frame m-1 Data)        | x Data bits for Frame (m-1) (14 bits Parity + Data) <sup>6</sup>   | Include: End      |
|  | (CRC[15..0])            | 16-bit CRC   | CRC               |
|  | 11111111                | Dummy byte (8 bits of ones)  | Start:<br>Include |
| Frame (Power Control)                  | 01010110                | LSC_POWER_CTRL Command   | Include           |
|  | 0                       | CRC Comparison Flag (0 = no)   | Include           |
|  | 000....0000             | Operand 22 zeroes  | Include           |
|  | X                       | Enable the VCCPG Power Domain (1= yes)   | Include           |
| Frame<br>(Init Bus Address)            | 11110110                | LSC_INIT_BUS_ADDR Command  | Include           |
|  | 0                       | CRC Comparison Flag (0 = no)   | Include           |
|  | 0000.....000000         | 23-bit Command Information (23 zeroes).  | Include           |
|  | 00                      | Default 2 zeroes (Reserved)  | Include           |
|  | Data[29:28]             | Bus Width <sup>5</sup><br>0: 8-bit Hard IP<br>1: 10-bit Hard IP<br>2: 10-bit EBR /Large RAM<br>3: 32-bit Hard IP | Include           |
|  | Data [27:17]            | ID CODE  | Include           |
|  | Data[16:0]              | Starting address   | Include           |
| Frame                                  | 01110010                | LSC_INIT_BUS_WRITE Command   | Include           |
|  | 1                       | CRC Comparison Flag (1 = yes) <sup>10</sup>  | Include           |

| Frame                                   | Contents[D7...D0] | Description   | CRC             |
|---|-------------------|---|-----------------|
| (Write data through Init Bus)           | 1                 | CRC Comparison at End Flag (1 = Execute CRC comparison at the end of all operands).   | Include         |
|   | 0                 | Exclude dummy bytes from bit stream.  | Include         |
|   | 1                 | Dummy definition (1 = use the next 4-bit Dummy Definition settings).  | Include         |
|   | 0000              | Dummy definition.   | Include         |
|   | (16-bit Size)     | Number of 72-bit frames in the operand. One = 0000000000000001.   | Include         |
|   | (Data)            | Data Frame  | Include: End    |
|   | (CRC[15..0])      | 16-bit CRC  | CRC             |
| Frame (SED CRC) OPTIONAL FRAME          | 10100010          | LSC_PROG_SED_CRC Command  | Start : Include |
|   | 0                 | CRC Comparison Flag (0 = no)  | Include         |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes).   | Include         |
|   | (CRC[31..0])      | 32-bit SED CRC  | Include         |
| Frame (Program Security) OPTIONAL FRAME | 11001110          | ISC_PROGRAM_SECURITY Command  | Include         |
|   | 0                 | CRC Comparison Flag (0 = no)  | Include         |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes).   | Include         |
| Frame (USERCODE)                        | 11000010          | ISC_PROGRAM_USRCODE Command   | Include         |
|   | 1                 | CRC Comparison Flag (1 = yes) <sup>10</sup>   | Include         |
|   | 0000.....000000   | 23-bit Command Information (23 zeroes).   | Include         |
|   | (U[31..0])        | 32-bit User code Data.  | Include: End    |
|   | (CRC[15..0])      | 16-bit CRC  | CRC             |
| Frame (Program Done)                    | 01011110          | ISC_PROGRAM_DONE  | Exclude         |
|   | 0                 | CRC Comparison Flag (0 = no)  | Exclude         |
|   | 00000....0000     | Operand 21 zeroes   | Exclude         |
|   | XX                | Behavior 00 = No overload(default) , 01 = Serial System Configuration Daisy Chain Support in Bit stream , 10 = Parallel System Configuration Daisy Chain Support in bit stream, 11 = No overload(default) | Exclude         |
| Frame (DUMMY)                           | 1111...1111       | 4 bytes DUMMY command (all ones).   | Exclude         |

**Notes:**

1. Data is byte bounded and must be padded with ones (1). For example, a 6-bit data frame of b111010 is written in the bitstream as b11111010 to make it byte bounded.
2. Only the data frame bits, highlighted in yellow, can be compressed.
3. If SED option is not chosen, place command DUMMY (NOOP) (32 bits of 1) on the command space. NOOP command is not included in CRC calculation. This is necessary to make the file size consistent with or without the security option you chose. It must be replaced with the LSC\_PROG\_SED\_CRC frame by Bitgen if you implement SED. Include in CRC calculation if LSC\_PROG\_SED\_CRC command. Do not include in CRC calculation if NOOP.
4. The CRC must be flipped so that CRC[0] is the LSB and CRC[15] is the MSB. For example, CRC = 0x0823 (CRC[15]...CRC[0]) in the bit file above is b1100010000010000. Refer to Appendix Generic Configuration Specification for CRC calculation details.
5. Insert a Write EBR Frame for each initialized EBR, or for each group of EBR Frames that is initialized to the same value. Omit if there is no EBR initialization data.
6. By default, this frame must be the Verify ID command. Bitgen shall have a switch to allow the Verify ID frame to be replaced with NOOP (all 1s). Include in CRC calculation if Verify ID command. Do not include in CRC calculation if NOOP.
7. If security option is not chosen, place command NOOP (32 bits of 1) on the command space. NOOP command is not included in CRC calculation. This is necessary to make the file size consistent with or without security option that you chose. Include in CRC calculation if Program Security command. Do not include in CRC calculation if NOOP.
8. The Control Register 0 must be set to the SW Default value, setting Control Register 0 bits [31..30] = [0..0].

## Read Back Bitstream Format

If required, Lattice Radiant generates a binary read back file. The read back file contains an ASCII comment string, device ID, control register 0 data, frame data, dummy bits, LUT, EBR, and usercode data. It does not contain a header, commands, or CRC. The data is ready to be shifted into the devices as required by the programming flow. The data is listed from MSB to LSB, where the MSB is the first bit shifted into TDI and the LSB the last bit. The MSB is also the first bit shifted out of TDO. The data for each frame is byte bounded. The byte must be padded with all ones (1). For example, a 6-bit data frame of b111010 would be written in the readback file as b11111010. The format is described in [Table B.4](#).

**Table B.4. Nexus Read Back File Format**

| Frame                   | Contents (D0..D7) | Description   |
|-------------------------|-------------------|---|
| Comments                | (Comment String)  | ASCII Comment (Argument) String   |
| Comment Terminator      | 11111110          | Read Back File Comment Terminator = 0xFE (binary file (.rbk) only)              |
| Device ID               | (ID[0..31])       | Expected 32-bit Device ID (See Note 1)  |
| Control Register 0      | (CtlReg0[0..31])  | Control Register 0 Data   |
| Device Specific Padding | 1111...1111       | Terminator bytes (all 1s). See <a href="#">Table B.5</a> for number of bytes.   |
| Frame 0                 | 1111...1111       | Dummy (Stop) bytes (all 1s). See <a href="#">Table B.5</a> for number of bytes. |
|                         | (Frame 0 Data)    | Data for Frame 0 (byte bounded, padded with zeros)                              |
| Frame 1                 | 1111...1111       | Dummy (Stop) bytes (all 1s). See <a href="#">Table B.5</a> for number of bytes. |
|                         | (Frame 1 Data)    | Data for Frame 1 (byte bounded, padded with zeros)                              |
| •                       | •                 | •   |
| •                       | •                 | •   |
| •                       | •                 | •   |
| Frame n-1               | 1111...1111       | Dummy (Stop) bytes (all 1s). See <a href="#">Table B.5</a> for number of bytes. |
|                         | (Frame n-1 Data)  | Data for Frame n-1 (byte bounded, padded with zeros)                            |
| Usercode                | (U[0..31])        | 32-bit Usercode Data  |

## Nexus Device Specific

**Table B.5. Nexus Device Specifics**

| Device Name           | Configuration Frames (n) | Byte Bound Padding Bits | Actual Bits per Frame (w) | Parity Bits (ECC) | CRC Bits | Dummy | Total Data Bits per Frame (x) |
|-----------------------|--------------------------|-------------------------|---------------------------|-------------------|----------|-------|-------------------------------|
| LIFCL-17<br>LFD2NX-17 | 4072                     | 0                       | 554                       | 14                | 16       | 8     | 592                           |
| LIFCL-40<br>LFD2NX-40 | 9172                     | 4                       | 662                       | 14                | 16       | 8     | 704                           |

**Table B.6. Nexus Device ID**

| Device Family | Device Name | 32-bit IDCODE |
|---------------|-------------|---------------|
| CrossLink-NX  | LIFCL-17    | 0x010F0043    |
|               | LIFCL-40-ES | 0x010F1043    |
|               | LIFCL-40    | 0x110F1043    |
| Certus-NX     | LFD2NX-17   | 0x310F0043    |
|               | LFD2NX-40   | 0x310F1043    |

## Technical Support Assistance

Submit a technical support case through [www.latticesemi.com/techsupport](http://www.latticesemi.com/techsupport).

## Revision History

### Revision 1.2, March 2021

| Section        | Change Summary   |
|----------------|--|
| All            | Changed CSNO and SDO to <i>MCSNO</i> and <i>MSDO</i> across the document.  |
| Features       | Updated content to change document title, number, and link of Advanced Security Encryption Key Programming Guide for Nexus Platform (FPGA-TN-02126) to <i>Advanced Configuration Security Usage Guide for Nexus Platform (FPGA-TN-02176)</i> . |
| Daisy Chaining | Updated <a href="#">Figure 9.3</a> and <a href="#">Figure 9.4</a> .  |

### Revision 1.1, June 2020

| Section                                    | Change Summary  |
|--|---|
| All  | <ul style="list-style-type: none"> <li>Changed document name from CrossLink-NX sysCONFIG Usage Guide to <i>sysCONFIG Usage Guide for Nexus Platform</i>.</li> <li>Added Nexus device across the document.</li> <li>Fixed formatting across the document.</li> </ul> |
| Features                                   | Updated section content.  |
| Configuration Details                      | <ul style="list-style-type: none"> <li>Updated Figure 4.1 in Configuration Ports Arbitration.</li> <li>Updated Table 4.1.</li> </ul>  |
| Master Port Configuration Process and Flow | Updated content of Early I/O Release.   |
| Configuration Modes                        | Updated Table 6.11, Table 6.13, Table 6.14, and Table 6.15.   |
| Software Selectable Options                | <ul style="list-style-type: none"> <li>Updated heading numbers.</li> <li>Updated Table 7.1 to remove 1.0 V, 1.2 V, and 1.5 V from the configuration port support.</li> <li>Removed 1.0 V, 1.2 V, and 1.5 V in CONFIG_IOVOLTAGE section.</li> </ul>                  |

### Revision 1.0, December 2019

| Section | Change Summary  |
|---------|-----------------|
| All     | Initial release |



[www.latticesemi.com](http://www.latticesemi.com)