MachXO5-NX Programming and Configuration User Guide

Technical Note

FPGA-TN-02271-2.1

July 2024

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, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. 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. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. 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

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
CRC	Cyclic Redundancy Check
EBR	Embedded Block RAM
ECDSA	Elliptic Curve Digital Signature Algorithm
HMAC	Hash-based Message Authentication Code
I^2C	Inter-Integrated Circuit; A synchronous, multi-master, multi-slave, packet switched, single-ended, serial bus
JEDEC	Joint Electron Device Engineering Council
JTAG	Joint Test Action Group
LMMI	Lattice Memory Mapped Interface
LSB	Least Significant Bit
LUT	Look Up Table
MIB	Memory Interface Block
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
SFDP	Serial Flash Discoverable Parameters
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 MachXO5™-NX family is based on the FDSOI technology. It reaps many benefits of the 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 MachXO5-NX device has many new and unique features that make it one of the best-in-class FPGA in its device density range. The MachXO5-NX 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 MachXO5-NX FPGA is built using volatile SRAM. Therefore, the on-chip flash 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 MachXO5-NX device when it powers up or anytime the device needs to be updated. The MachXO5-NX device provides a rich set of features for configuring the FPGA or programming the on-chip non-volatile memory. There are many options available for building the programming solution that fits your needs. Each of the options available is 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 the MachXO5-NX Family Datasheet (FPGA-DS-02102).

For information regarding MachXO5-NX Root-of-Trust devices, refer to the MachXO5-NX Family Root-of-Trust Devices Datasheet (FPGA-DS-02120), MachXO5-NX Root-of-Trust Device Provisioning User Guide (FPGA-TN-02333), Embedded Security and Function Block User Guide for MachXO5-NX (FPGA-TN-02320), and Embedded Security and Function Block with Advanced Key Management for MachXO5-NX (55TD) Devices (FPGA-TN-02353). This technical note only includes information relevant to MachXO5-NX Root-of-Trust devices in the Bitstream Sizes section, Table B.5, and Table B.6.

2. Features

The following lists the key programming and configuration features of the MachXO5-NX device:

- Ultrafast I/O configuration for instant-on support
- Fast full device configuration
- I3C device configuration support
- Bitstream dry-run support to ensure bitstream integrity
- Extremely low Soft Error Rate (SER) due to inherent FDSOI technology
- Enhanced and flexible multi-boot support (32-bit addressing support with jump-table support)
- Configuration bridging for easy on-chip flash programming
- User-selectable Booting Sequence
- Bitstream Encryption/Decryption – 256 bits AES
- Bitstream Authentication – HMAC
- Bitstream Authentication – ECDSA
- Multiple programming and configuration interfaces:
  - 1149.1 JTAG
  - Self-download
  - Slave I^2C (For SRAM Configuration)
  - Slave I3C (For SRAM Configuration)
  - Slave SPI, includes SERIAL, DUAL, and QUAD mode
  - Dual Boot and Multi-Boot support
  - Ping-Pong Boot
  - Daisy chaining
- Transparent programming of on-chip flash 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 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 on-chip flash or other memory device. It is a binary file and is programmed unmodified into the on-chip flash memory 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 MachXO5-NX device include JTAG, SPI, I^2C, and I3C physical connections.
- User Mode – The MachXO5-NX device is in user mode when configuration is complete, and the FPGA is performing the logic functions that 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 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 tri-stated.
- Dual Boot – Supports two configuration patterns that reside in the flash memory. Whenever loading failure occurs with the primary pattern, the FPGA searches for and loads a so-called *golden* or fail-safe pattern. Both patterns come from the on-chip flash memory.
- Ping-Pong Boot – Utilize the jump table to select an image for booting without changing location of the image in the flash memory.
- Multi Boot – The FPGA 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 to choose to load on demand. All the patterns are stored in an on-chip flash memory.
- Transparent Mode – Transparent mode, also referred to as Background Mode, is used to update the SRAM configuration while leaving the MachXO5-NX 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
- Serial Peripheral Interface (SPI) 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.
- 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 cyclic redundancy check (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 MachXO5-NX 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 internal flash memory. The non-volatile memory holds the configuration data that is loaded into the FPGAs SRAM.

4.1. Bitstream Sizes

The MachXO5-NX devices are SRAM-based FPGAs. The SRAM configuration memory must be loaded from the non-volatile memory that can store all 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 MachXO5-NX 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		All Uncompressed
	Scenario	Unencrypted/Encrypted Bitstream Size (Mb)
LFMXO5-25	No LRAM, No EBR,	5.377
	No LRAM, MAX EBR	6.817
	MAX LRAM, No EBR	5.877
	MAX LRAM, MAX EBR	7.317
LFMXO5-15D	MAX LRAM, MAX EBR	7.362
LFMXO5-55T	No LRAM, No EBR,	15.005
	No LRAM, MAX EBR	18.749
	MAX LRAM, No EBR	18.589
	MAX LRAM, MAX EBR	22.333
LFMXO5-100T	No LRAM, No EBR,	15,005
	No LRAM, MAX EBR	18,749
	MAX LRAM, No EBR	18.589
	MAX LRAM, MAX EBR	22.333
LFMXO5-55TD	MAX LRAM, MAX EBR	22.431

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. MachXO5-NX 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)
	Slave I^2C	Slave I^2C 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 auto booting mode to boot the device from on-chip flash memory, 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 auto-booting event, and allows the slave configuration ports (Slave SPI or Slave I^2C) 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 auto booting sequence.

The PROGRAMN pin becomes a user I/O pin through a user feature setting, with default PROGRAMN pin state high to achieve a SDM 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 in Table 4.3.

Table 4.3. Slave Configuration Port Activation Key

Slave Port / Activation Key	Slave Configuration Port Activation Key	
Slave SPI Port	Dummy Bytes[1]	32'HA4C6F48A
Slave I^2C/I3C Port	Slave I^2C/I3C Port Address[2]	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^2C address could be either 7 bits or 10 bits address.

The device configuration control flow of CFG_TOP is shown in Figure 4.1.

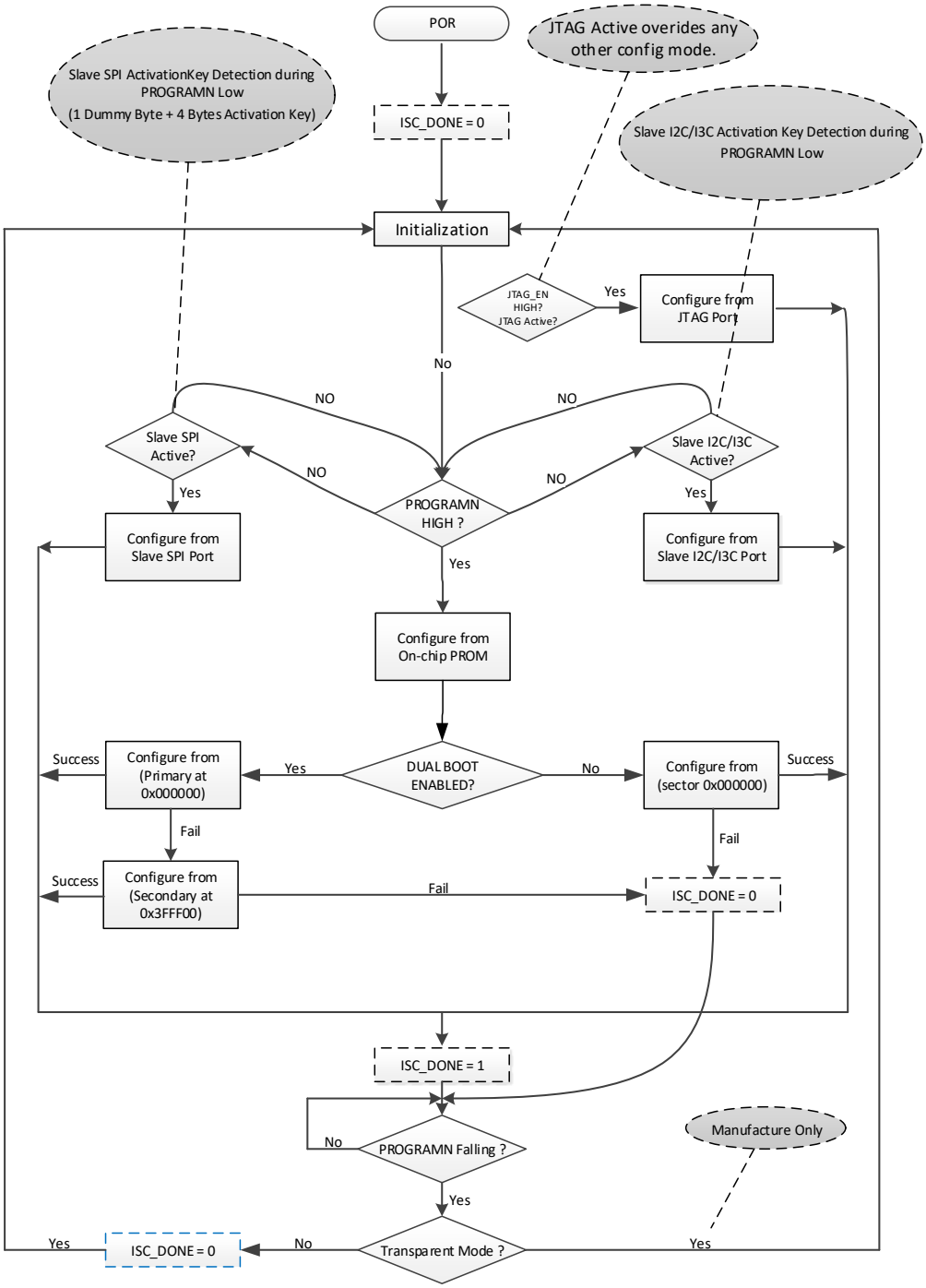


Figure 4.1. Configuration Control Flow

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

4.4. sysCONFIG Pins

The MachXO5-NX device provides a set of sysCONFIG I/O pins for you to program and configure the FPGA. The sysCONFIG pins are grouped together to create ports (such as JTAG, SSPI, and I²C/I3C) 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 the Global tab.
- You must prevent external logic from interfering with the 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 MachXO5-NX device, the default state of the SSPI sysCONFIG pins become general-purpose I/O. This means that you lose the ability to program the MachXO5-NX device using SSPI or Slave I²C/I3C when using the default sysCONFIG port settings. To retain the SSPI or Slave I²C/I3C 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 VCCIO1 and VCCIO2 voltage. It is crucial for this to be taken into consideration when provisioning other logic attached to Bank 1 and Bank 2.

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	SSPI	I²C/I3C
JTAG_EN	Dedicated	DOWN	1'b1	1'b0	1'b0
PROGRAMN	Shared	UP	1'b0	1'b0	1'b0
INITN	Shared	UP	INITN		
DONE	Shared	UP	DONE		
MCSNO	Shared	UP	—	—	—
TCK/SCLK	Shared	UP	TCK	SCLK	—
TMS/SCSN[1]	Shared	UP	TMS	SCSN[1]	—
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
SD[15:4][2]	Shared	UP	—	—	—

Notes:

1. SCSN should have 4.7 kΩ pull-up resistor on-board for SSPI.
2. Reserved when the device is in configuration mode.

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. Holding the PROGRAMN pin low prevents the MachXO5-NX device from leaving the Initialization phase. The PROGRAMN has a minimum pulse width assertion period ($t_{PROGRAMN}$) for it to be recognized by the FPGA. You can find this minimum time in the sysCONFIG Port Timing Specifications of the MachXO5-NX Family Datasheet (FPGA-DS-02102).

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

- If the device is currently being programmed through 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 VCCIO1 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 $t_{PROGRAMN}$ 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 must be driven high before the ISC_DISABLE command is issued.

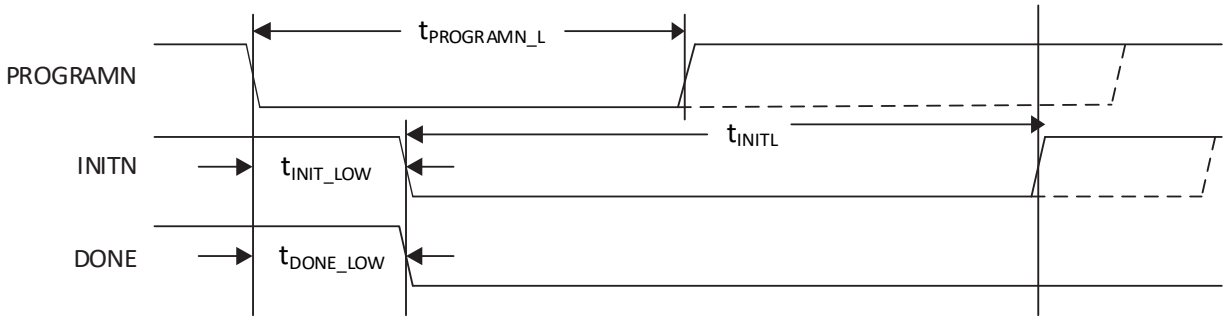


Figure 4.2. Configuration from PROGRAMN Timing

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_{INIT_LOW} parameter.
- After the t_{INITL} period has elapsed, the INITN pin is deasserted (that is active high) to indicate the MachXO5-NX device is ready for its configuration bits. The MachXO5-NX device begins loading configuration data from the on-chip flash memory.
- INITN can be asserted low by an external agent before the t_{INITL} period has elapsed 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_{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_{INITL} has elapsed and the INITN pin has gone high, any subsequent INITN assertion signals the MachXO5-NX device has detected an error during configuration.

The following conditions cause INITN to become active, indicating that 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.

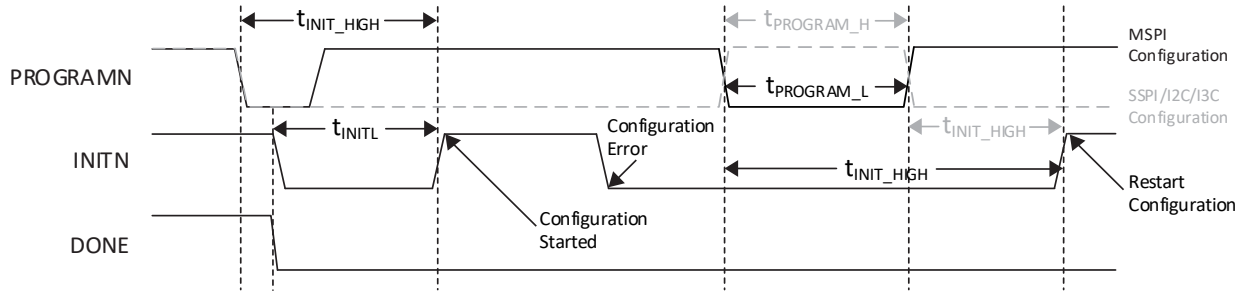


Figure 4.3. Configuration Error Notification

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 on-chip flash. This can occur when the device is in self-download configuration mode and the on-chip flash memory is not programmed.
- The program done command is not received when the end of on-chip SRAM configuration or on-chip Flash memory is reached.

Note: When INITN is low, JTAG and SPI have full access to the MachXO5-NX device configuration engine and internal flash memory.

Perform the following steps after a configuration error to enable the device to restart configuration:

1. After INITN goes low due to a configuration error, toggle the PROGRAMN pin. You can find the minimum pulse width for $t_{PROGRAMN\_H}$ and $t_{PROGRAMN\_L}$ in the sysCONFIG Port Timing Specifications of the MachXO5-NX Family Datasheet (FPGA-DS-02102).
2. Wait for t_{INIT_HIGH} time period to elapse. The INITN pin de-asserts high indicating that the Nexus device is ready to restart configuration.

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. After the internal DONE bit is asserted the GPIO will wake-up, the external DONE pin goes high after 10 µs. This means that the GPIO pin settings are applied before the DONE pin goes high.

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 through 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. Slave sysCONFIG Pins

4.5.1. 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 MachXO5-NX Family Datasheet (FPGA-DS-02102). 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 MachXO5-NX 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 sysCONFIG Port Timing Specifications of the MachXO5-NX Family Datasheet (FPGA-DS-02102).

4.5.2. 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.

4.5.3. 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.

4.5.4. 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.

4.5.5. SD2/SCL

This pin is used by both the Slave SPI and Slave I²C/I3C configuration interface.

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

SCL – In Slave I²C/I3C configuration mode, this is the serial clock line of the I²C or I3C bus.

4.5.6. SD3/SDA

This pin is used by both the Slave SPI and Slave I²C/I3C configuration interface.

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

SDA – In Slave I²C/I3C configuration mode, this is the serial data line of the I²C or I3C bus.

4.6. PERSISTENT

The internal PERSISTENT control bits are used to determine whether the dual-purpose Slave sysCONFIG pins remain as sysCONFIG pins during normal operation, for example, to support transparent programming or configuration. The MachXO5-NX device has several PERSISTENT physical SRAM cells that determine the existence of the Slave SPI port or I²C/I3C port after entering user mode. These settings can be set in Lattice Radiant Device Constraint Editor under the Global tab as shown in Table 4.5.

Table 4.5. sysCONFIG Pins Global Preferences

Port Setting	Value	Pins Affected	Details
SLAVE_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_I2C_PORT/ SLAVE_I3C_PORT	ENABLE	SCL, SDA	If enabled, persisted for configuration purpose.

Note: JTAG Persist follows JTAG enable pin.

5. Configuration Process and Flow

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

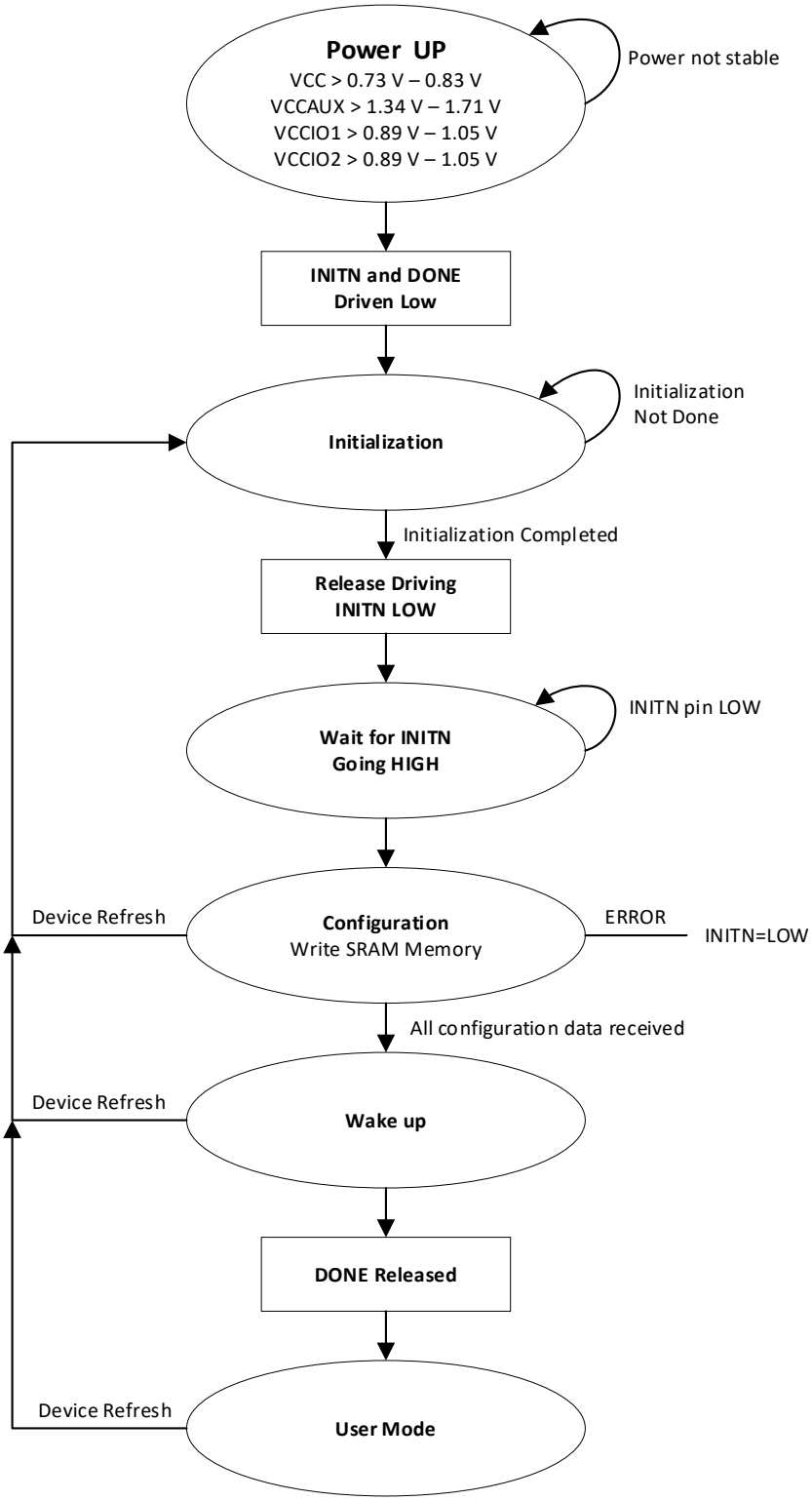


Figure 5.1. Configuration Flow

The MachXO5-NX 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 MachXO5-NX device. The Configuration Ports Arbitration section provides information about the availability of each sysCONFIG port.

5.1. Power-up Sequence

For the MachXO5-NX 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. During the power-up sequence phase, once the POR circuit is active, the POR circuit makes sure that the external I/O pins are in a high impedance state. It also monitors the VCC, VCCAUX, VCCIO1, and VCCIO2 input rails. The POR circuit waits for the following conditions:

- VCC > 0.73 V – 0.83 V
- VCCAUX > 1.34 V – 1.71 V
- VCCIO1 > 0.89 V – 1.05 V
- VCCIO2 > 0.89 V – 1.05 V

When these conditions are met the POR circuit releases an internal reset strobe, allowing the device to begin its initialization process. The MachXO5-NX 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.

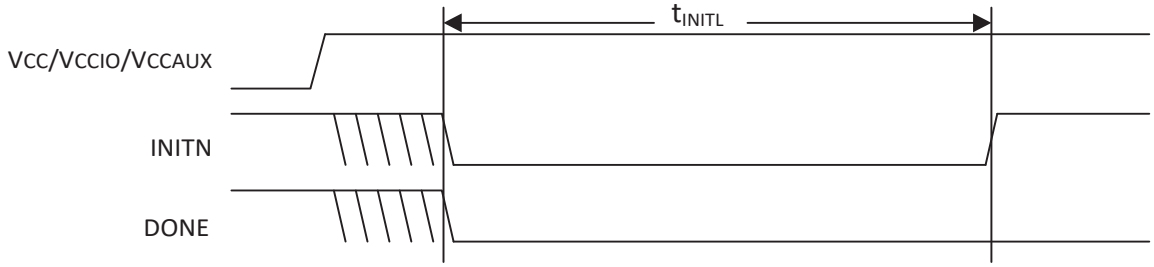


Figure 5.2. Configuration from Power-On-Reset Timing

5.2. Initialization

The MachXO5-NX 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 the SRAM memory inside the FPGA.

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

- The t_{INITL} 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} period, the FPGA is clearing the configuration SRAM. When the MachXO5-NX device is part of a chain of devices each device has different t_{INITL} initialization time. 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_EN 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 can accept the configuration bitstream created by the Lattice Radiant software.

For proper bitstream data alignment, the bitstream Preamble must be detected once. If the preamble does not come before the preamble timer (counting for 600,000 SPI clock cycles) expires, a boot failure is declared, and the boot process aborts.

Once the preamble is detected, the MachXO5-NX device continues fetching data from non-volatile memory to configure the FPGA SRAM memory. The MachXO5-NX 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 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 MachXO5-NX device's fixed four-phase wake-up sequence starts when the device has correctly received all 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 those sequences four controls. The four control strobes are:

- Global Set/Reset (GSRN)
- Global Write Enable (GWE)
- Global Output Enable (GOE)
- External DONE

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 (GSRN).

Other phases of the Wake-Up process release the Global Set/Reset and the Global Write Enable 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 Enable 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 GWE deasserted 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 MachXO5-NX 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 Bank 1, I/O Bank 2, I/O Bank 6, and I/O Bank 7), to assume user-defined drive states at the beginning of bitstream 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 the left/right Memory Interface Block (MIB) 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 users 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 MachXO5-NX device, the Early I/O Release feature is not supported.

5.6. User Mode

The MachXO5-NX device enters user mode immediately following the Wake-Up sequence has completed. User Mode is the point in time when the MachXO5-NX device begins performing the logic operations that you have designed. The device remains in this state until one of three events occurs:

- The PROGRAMN input pin is asserted.
- A REFRESH command is received through 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 MachXO5-NX 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 MachXO5-NX 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 MachXO5-NX device to drive INITN and DONE low. The MachXO5-NX 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 MachXO5-NX 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 MachXO5-NX 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. Self-Download Mode

The advantages of Self-Download Configuration Mode include:

- Speed: The MachXO5-NX device is ready to run in a few milliseconds depending on the density of the device.
- Security: The configuration data is never seen outside the device during the load to SRAM. You can prevent the internal memory from being read.
- Reduced cost: There is no need to purchase a PROM specifically reserved for programming the MachXO5-NX.
- Reduced board space: Elimination of an external PROM allows the board to be smaller.

The MachXO5-NX device retrieves the configuration data from the internal flash memory when it is using Self-Download Mode. SDM is triggered when power is applied, a REFRESH command is received, or by asserting the PROGRAMN pin. The Self-Download Mode cannot be used when the configuration memory overflow occurs. Other configuration modes must be used in the event of the memory overflow.

6.2. Slave SPI Mode

The MachXO5-NX device provides a Slave SPI (SSPI) configuration port that allows users 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 on-chip flash memory updates are done using either offline or transparent operations. It is necessary to send a REFRESH command to load a new flash image into the configuration SRAM. When the MachXO5-NX 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 MachXO5-NX 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 tri-stated. Commands can be written into and data read from the MachXO5-NX 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 (SERIAL, DUAL or QUAD) for SLAVE_SPI_PORT in the Lattice Radiant Device Constraint Editor.

Using the SSPI port simplifies the MachXO5-NX device configuration process. Lattice provides *C* source code called sspiembedded to insulate users from the complexity of programming the MachXO5-NX device. Refer to the Lattice Radiant online help about sspiembedded.

With SSPI persistence enabled in user mode, there is a limitation on operation of the SSPI/JTAG bus when using the Flash Access IP for MachXO5-NX. Do not access the SSPI/JTAG port when accessing flash memory using the Flash Access IP or vice versa.

Table 6.1. 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 MachXO5-NX device Configuration Logic.
SCSN[1]	CSN	Input with weak pull-up	MachXO5-NX 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. 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 MachXO5-NX device Configuration Logic.
	D0	Inout	D0 of the data bus for DUAL and QUAD mode.
TDO/SO/SD1	MISO	Output	Carries output data from the MachXO5-NX Device Configuration Logic to the external SPI master. It is normally tri-stated 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:

1. Use external 4.7 kΩ pull-up resistor.

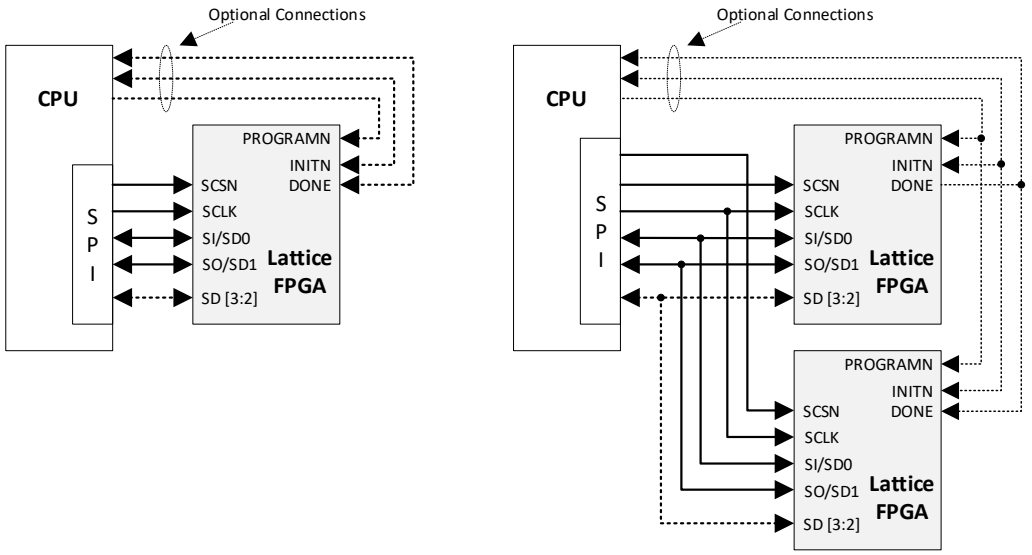


Figure 6.1. MachXO5-NX Slave SPI Port with CPU and Single or Multiple Devices

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.

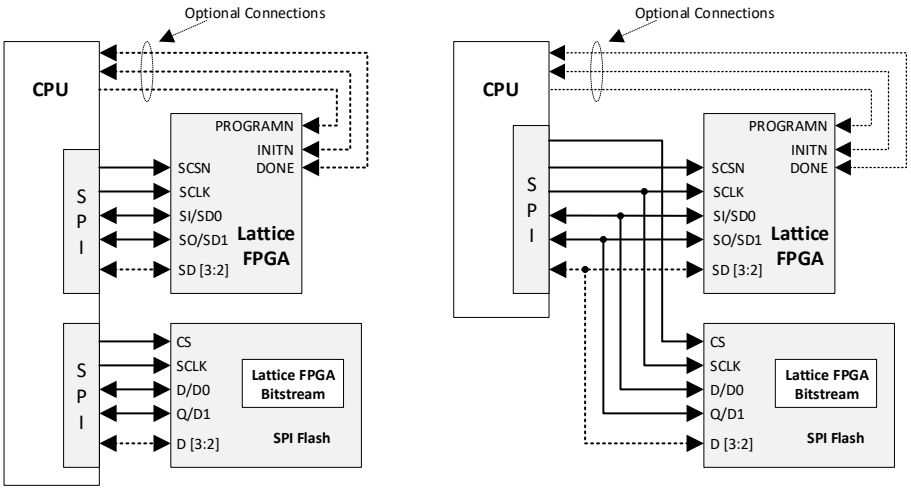


Figure 6.2. MachXO5-NX Slave SPI Port with SPI Flash

Notes:

- The dotted lines indicate the connection is optional.
- The MachXO5-NX 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 postpone the SDM 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 6.2. After the Slave SPI Configuration Port is activated, the state of the PROGRAMN pin is irrelevant.

Table 6.2. Slave SPI Configuration Port Activation Key

Slave Port/ Activation Key	Slave Configuration Port Activation Key	
Slave SPI Port	Dummy Bytes[1]	32'HA4C6F48A

Note:

1. 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 (SERIAL, DUAL or QUAD) for SLAVE_SPI_PORT in the 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 6.3 and Figure 6.4.

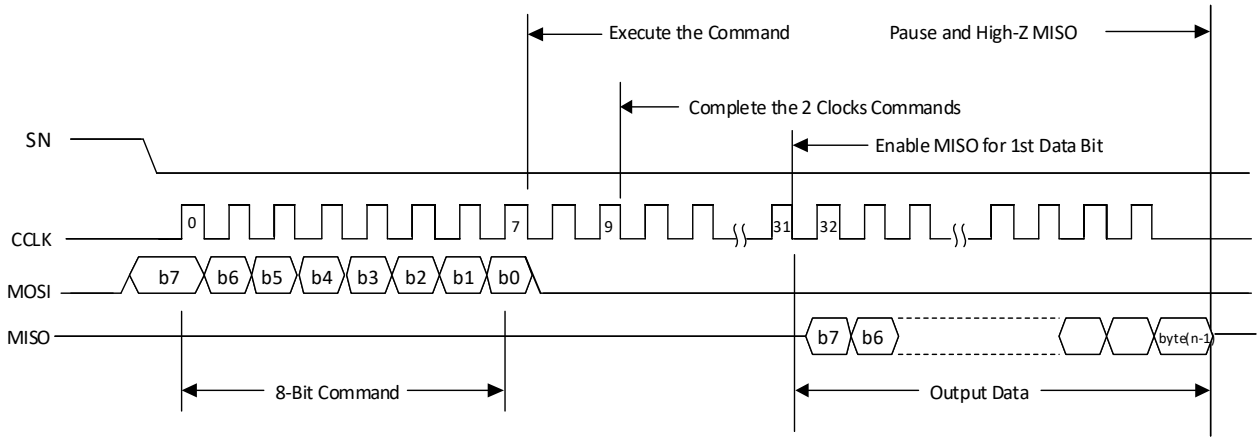
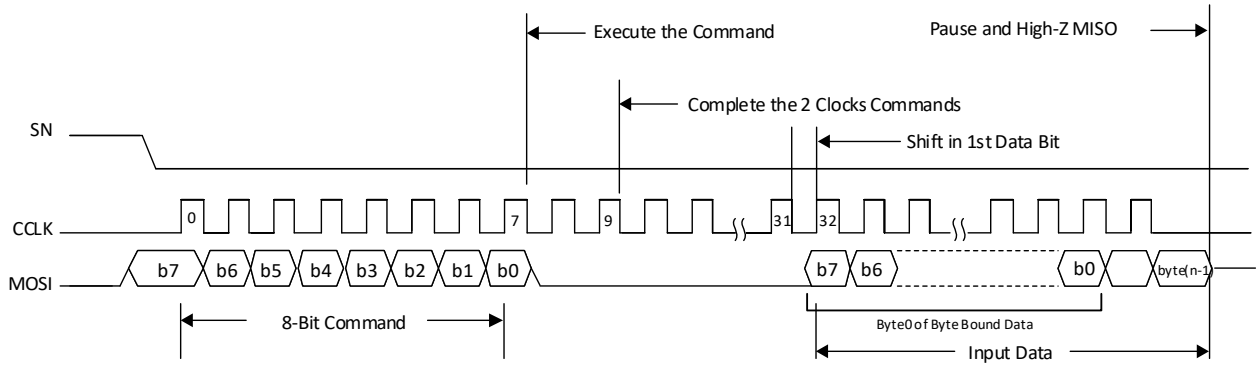


Figure 6.3. Slave SPI Read Waveforms



Note: The bitstream is transferred starting with the first byte of the data file, starting with the MSB of the byte.

Figure 6.4. Slave SPI Write Waveforms

6.2.3. Slave SPI Port AC Timing Requirements

The Slave SPI port maximum operation frequency requirement is shown in Table 6.3.

Table 6.3. Slave SPI Port AC Timing Requirements

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

For more information on the AC timing requirement for the MachXO5-NX Slave SPI configuration port, refer to the sysCONFIG Port Timing Specifications of the MachXO5-NX Family Datasheet (FPGA-DS-02102).

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 MachXO5-NX 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 6.4.

Table 6.4. 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 must be shifted 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. After 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 MachXO5-NX device bitstream and the encrypted bitstream (SSPI Mode).

The Slave SPI configuration flow diagram is shown in Figure 6.5. The following lists the highlights of the flow:

- 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 MachXO5-NX device is capable to wake up the I/O X, Y to user specified value at the beginning of the bitstream.
- The MachXO5-NX 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.

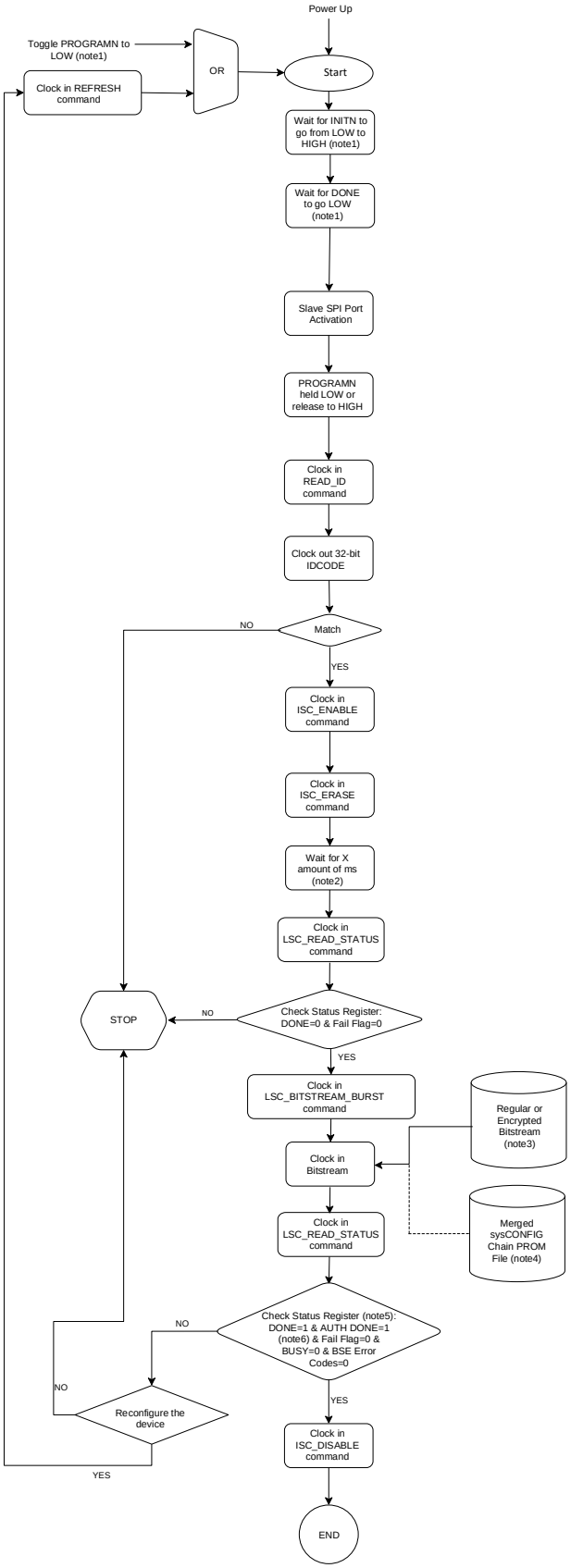


Figure 6.5. Slave SPI Configuration Flow

Notes:

1. For more information about the behavior of these pins, refer to the PROGRAMN, INITN, and DONE section. For more information about the timing specifications, refer to the MachXO5-NX Family Data Sheet.
2. Refer to Table 6.5 for X-amount of ms execution time per MachXO5-NX device.
3. For a single MachXO5-NX device, the input file is a bitstream, which may be a standard or encrypted bitstream.
4. For a sysCONFIG chain of devices, the input file can be a merged PROM file.
5. The external DONE pin toggles high after the DONE bit and AUTH DONE bit (if applicable) in the status register are updated. If the external DONE pin is not available for monitoring, after clocking in the bitstream, observe the following wait times before checking the status register: 102.06 ms (encrypted bitstream) or 60 µs (regular bitstream). The status register can be checked at any time, but the DONE bit and AUTH DONE bit (if applicable) will only be updated after the specified wait time.
6. The AUTH DONE=1 check is only applicable if the encrypted bitstream is used.

Table 6.5. Execution Time for ISC_ERASE Command

Device Name	Fixed Execution Time Value (ms)
LFMXO5-25	1.92
LFMXO5-55T	4.87
LFMXO5-100T	4.87

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 MachXO5-NX device. MSB of the byte bound 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.

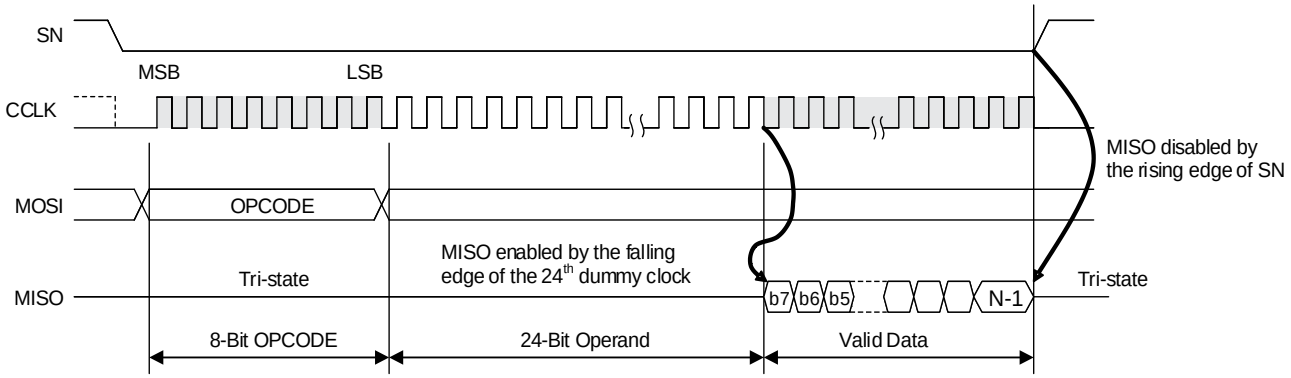


Figure 6.6. Class A Command Waveforms

6.2.6.2. Class B Command Waveforms

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

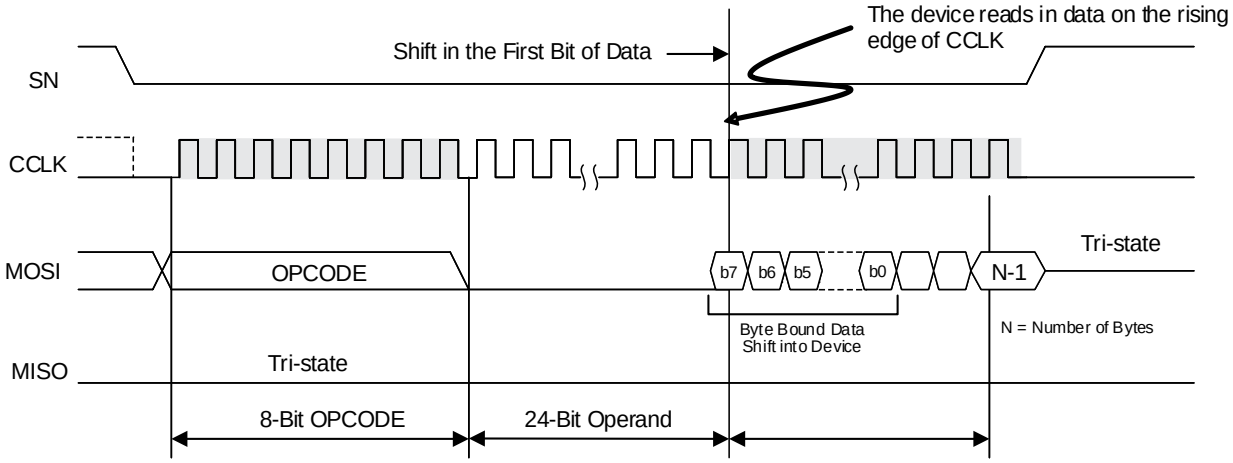


Figure 6.7. Class B Command Waveforms

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.

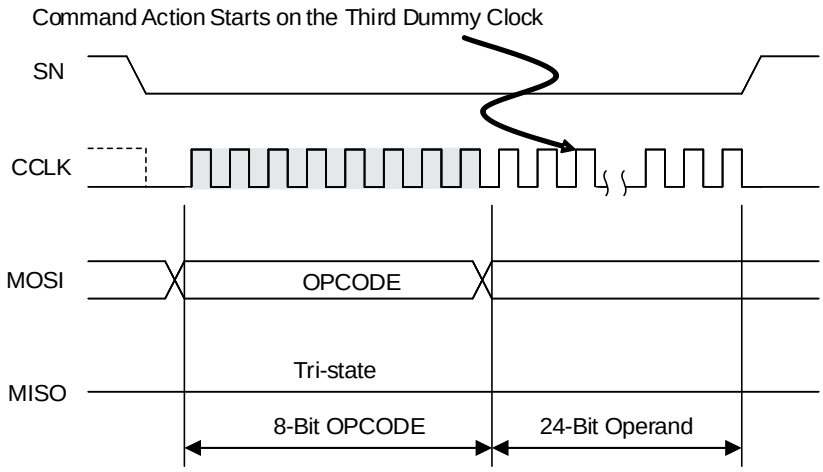


Figure 6.8. Class C Command Waveforms

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.

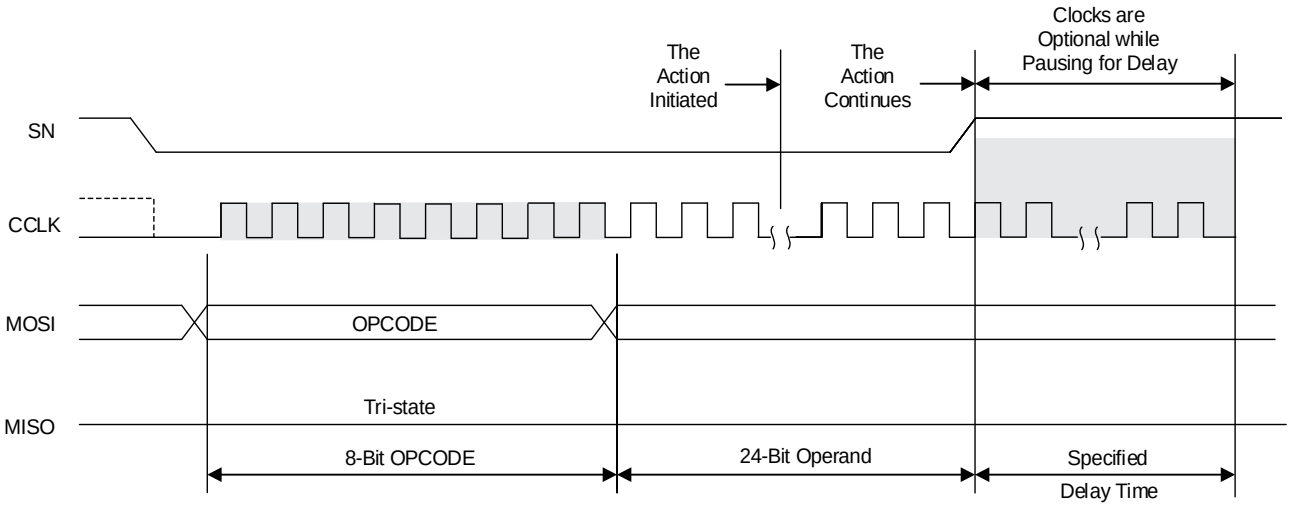


Figure 6.9. Class D Command Waveforms

6.3. Slave I²C/I3C Mode

The MachXO5-NX device provides a Slave I²C/I3C configuration port that allows users 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 MachXO5-NX 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 6.6 is supported.

In the Slave I²C/I3C mode, the SD2/SCL pin becomes SCL (that is, Serial Clock) of the I²C/I3C bus, and the SD3/SDA becomes the SDA (such as Serial Data) of the I²C/I3C bus.

The Slave I²C/I3C port can be activated through the CFG port arbitration procedure before the device enters user mode. The I²C/I3C port can be persisted in user mode by setting the desired Value for SLAVE_I2C_PORT/SLAVE_I3C_PORT in the Lattice Radiant Device Constraint Editor.

Using the I²C port simplifies the MachXO5-NX device configuration process. Lattice provides the *C* source code called i2cembedded to insulate you from the complexity of programming the MachXO5-NX device. Refer to the Lattice Radiant online help about i2cembedded.

Table 6.6. Slave I²C/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 MachXO5-NX device I²C/I3C configuration setup is shown in Figure 6.10.

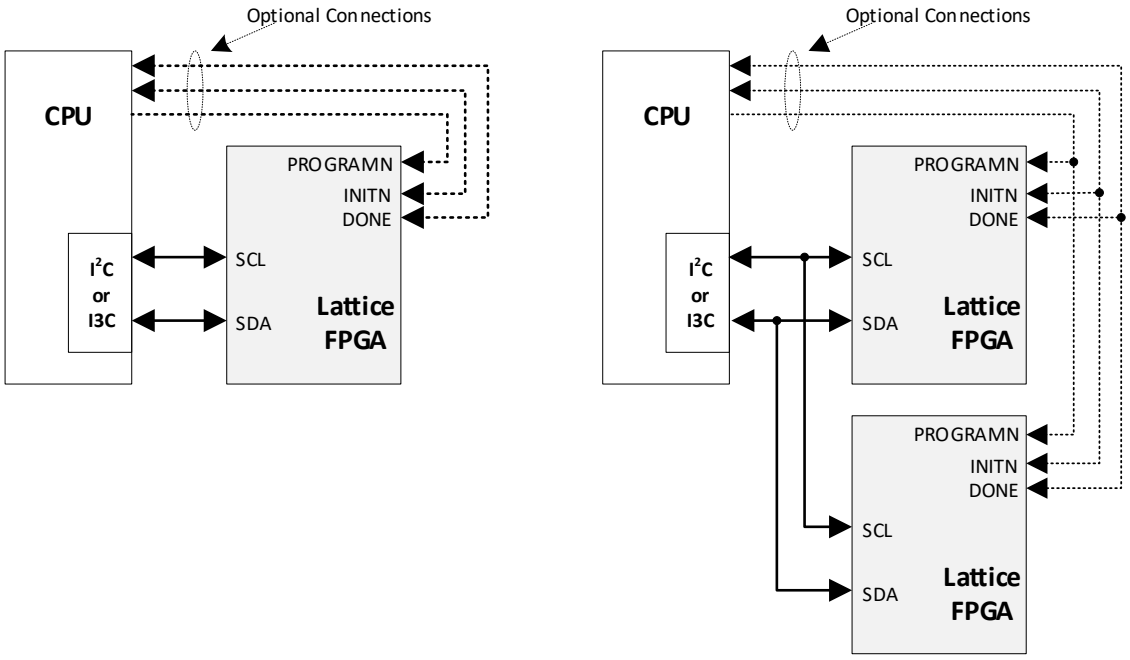


Figure 6.10. MachXO5-NX Slave I²C/I3C Port with CPU and Single or Multiple Devices

Note: For more details about the slave I2C/I3C configuration flow, refer to the slave I2C/I3C configuration flow diagram in the sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099).

6.3.1. Bus Sharing Between I²C and I3C

The MachXO5-NX device configuration block support both slave I²C and slave I3C interface, the two share the same bus as shown in Figure 6.11.

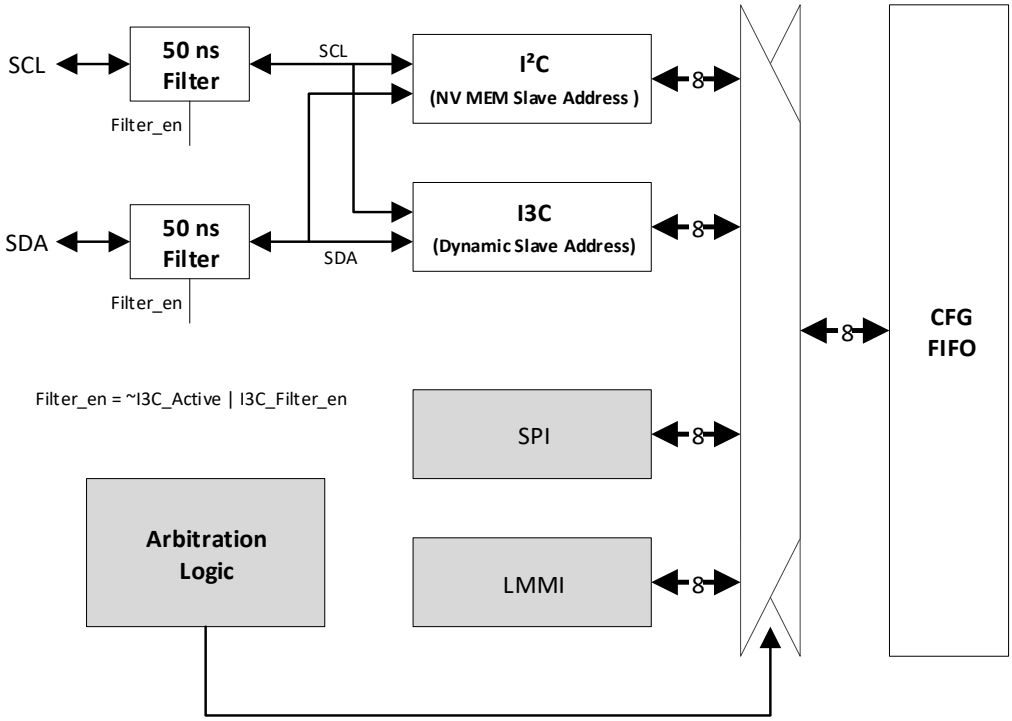


Figure 6.11. Bus Sharing between I²C and I3C

6.3.2. Slave I²C/I3C Configuration Mode

From configuration perspective, the I²C or I3C interface only support slave mode. They are for configuration purpose only, not usable by user logic. This hard slave I²C block and I3C block for configuration provide the industry standard two-pin interface for I²C/I3C masters to communicate with the MachXO5-NX device configuration engine.

6.3.2.1. Slave I²C Port Capabilities

- Standard I²C bus protocol to communicate with 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 filters for SCL and SDA make it capable of working within newer I3C bus.

6.3.2.2. Slave I3C Port Capabilities

- Two wire serial interfaces up to 12.5 MHz using Push-Pull.
- Legacy I²C Device co-existence on the same Bus.
- Dynamic Addressing while supporting Static Addressing for Legacy I²C support.
- Legacy I²C messaging.
- I²C-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 I²C and I3C Ports

6.3.3.1. Slave Address for I²C Configuration Port

An external I²C master accesses the Configuration Logic using address 1000000 (7-bit mode) or 1111000000 (10-bit mode) as default unless the CFG I²C base address has been modified. The CFG I²C slave address could be altered by 8 bits user programmable EFUSE bits (YYYXXXXX) inside Feature Row. Using Program File Utility – Feature Row Editor, the user specified I²C slave address becomes XXXXX00 for 7-bit mode and YYYXXXXX00 for 10-bit address.

6.3.3.2. Slave Address for I3C Configuration Port

For Legacy I²C support, the MachXO5-NX device allows users to program 8 bits EFUSE bit (AAABBBBB) to setup the static address for the I3C configuration port, with BBBBB00 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 I²C/I3C Port

Similar to all configuration ports, the Slave I²C or I3C port is enabled by the two standard methods:

- Enable the Slave I²C 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 postpone the SDM auto booting event. Then the Master performs the write activity to the slave address of the MachXO5-NX Slave I²C or I3C configuration port along with the Slave Configuration Port Activation Key, as shown in Table 6.7. After the Slave I²C or I3C Configuration Port been activated, the state of the PROGRAMN pin is irrelevant.

Table 6.7. Slave I²C/I3C Configuration Port Activation Key

Slave Port/ Activation Key	Slave Configuration Port Activation Key	
Slave I²C/I3C Port	Slave I²C/I3C Port Address[1]	32'HA4C6F48A

Note:

1. The slave I²C/I3C address could be either 7 bits or 10 bits address.

- Enabling Slave I²C/I3C port persistence in user mode.

  The I²C/I3C port could be persisted in user mode by setting the desired Value (ENABLE) for SLAVE_I2C_PORT/SLAVE_I3C_PORT in the Lattice Radiant Device Constraint Editor. Once set, the configuration bitstream contains optional I²C/I3C persistence bits. When the device completes configuration and wakes up, it checks the persistent bits to determine if the SLAVE I²C/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 I²C 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 I²C Port Activation Flow

The standard Slave I²C Configuration Port activation flow is shown in Figure 6.12.

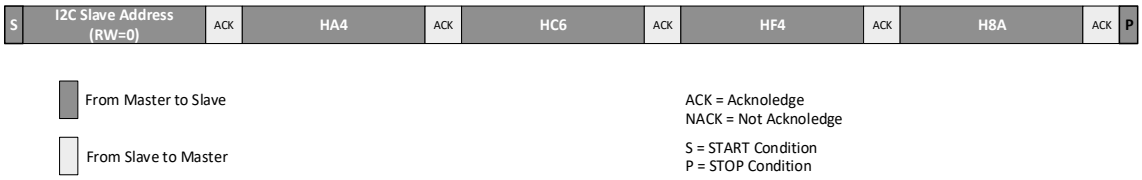


Figure 6.12. Slave I²C Configuration Activation Flow

6.3.4.2. Slave I3C Port Activation Flow

The typical Slave I3C Configuration Port activation flow is shown in Figure 6.13.

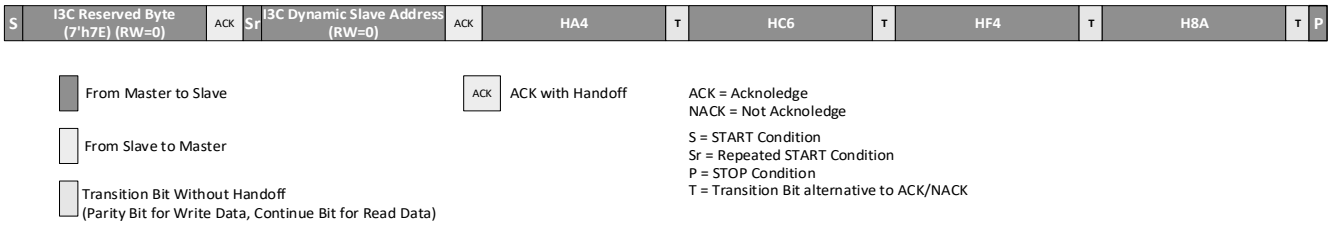


Figure 6.13. Slave I3C Configuration Activation Flow

6.3.5. Slave I²C/I3C Configuration Logic Access

The Slave I²C Configuration Port or Slave I3C Configuration Port follows corresponding standard to access the MachXO5-NX device by non-JTAG configuration engine, using the same non-JTAG command set, as shown in **Error! Reference source not found.**.

6.3.5.1. Slave I²C Configuration Access Flow

The typical I²C configuration write and read flows are shown in Figure 6.14 and Figure 6.15.

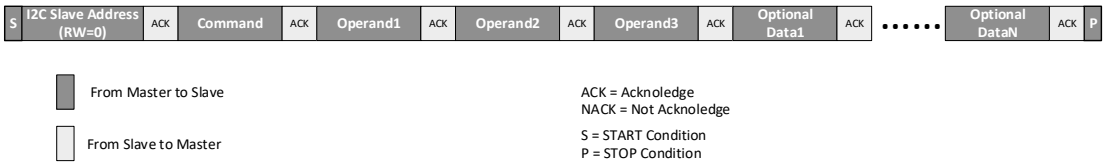


Figure 6.14. Typical I²C Write

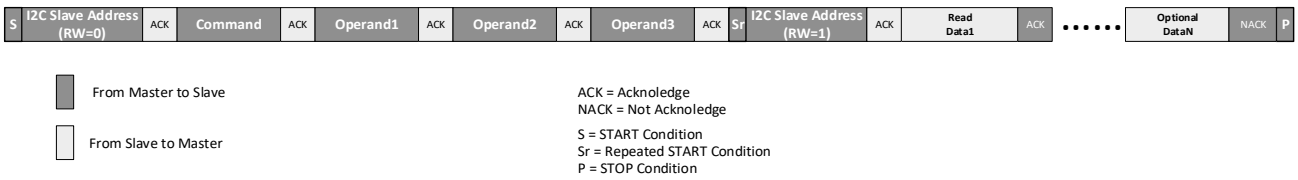


Figure 6.15. Typical I²C Read

6.3.5.2. Slave I3C Configuration Access Flow

The typical I3C configuration write and read flows are shown in Figure 6.16, Figure 6.17, Figure 6.18, and Figure 6.19.

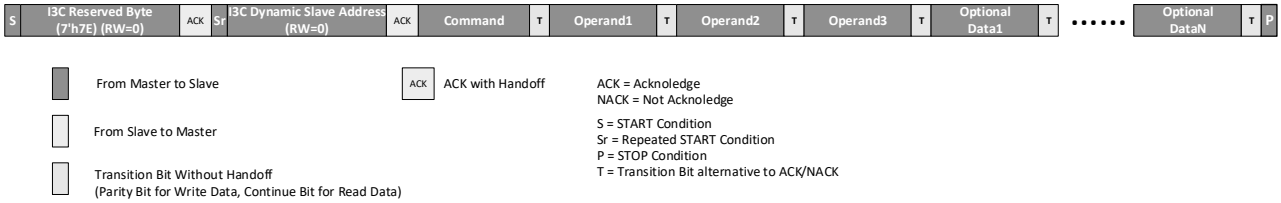


Figure 6.16. Typical I3C Write with 7'h7E

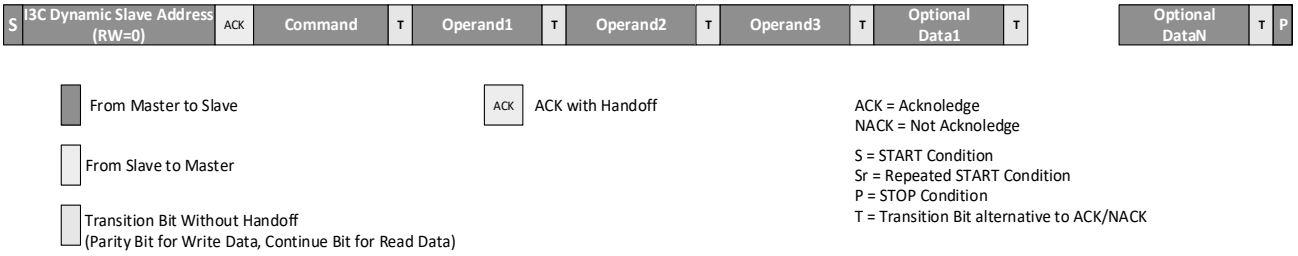


Figure 6.17. Typical I3C Write without 7'h7E

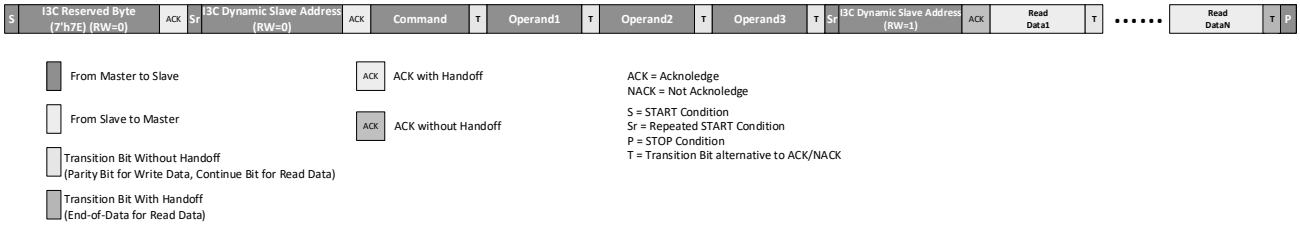


Figure 6.18. Typical I3C Read with 7'h7E

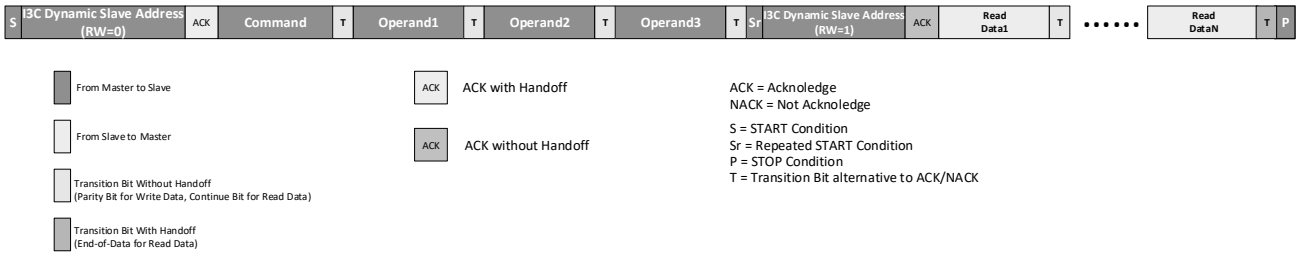


Figure 6.19. Typical I3C Read without 7'h7E

6.3.6. Slave I²C/I3C AC Timing Requirements

The Slave I²C and I3C maximum operation frequency requirements are listed in Table 6.8.

Table 6.8. Slave I²C/I3C Maximum Frequency Requirements

Description	Parameter	Min	Max	Unit
I²C Maximum SCL Clock Frequency	F_{SCL_I2C}	—	1	MHz
I3C Maximum SCL Clock Frequency	F_{SCL_I3C}	—	12	MHz

For the detailed AC timing requirements for the MachXO5-NX device Slave I²C/I3C configuration port, refer to the sysCONFIG Port Timing Specifications of MachXO5-NX Family Datasheet (FPGA-DS-02102).

6.4. JTAG Mode

The MachXO5-NX 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_EN pin can enable the JTAG port at any time. The JTAG port on MachXO5-NX devices provides:

- Flash memory programming
- Direct SRAM configuration
- Full access to the MachXO5-NX 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 MachXO5-NX 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 users to analyze the logic inside the MachXO5-NX device in the same fashion as an external logic analyzer permits analysis of board level logic. The Reveal tool access is only available through the JTAG port.
- SRAM Readback – The JTAG port can directly access the configuration SRAM. It is occasionally necessary to perform failure analysis for SRAM based FPGAs. A key component for failure analysis is 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 MachXO5-NX device on the Lattice website.

6.4.1. Method to Enable the JTAG Port

The four pins for the JTAG port on MachXO5-NX devices are dual-purpose I/O, which are shared with user functionalities. Once the dedicated JTAG_EN 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 6.9.

Table 6.9. JTAG AC Timing Requirements

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

For detailed AC timing requirements for the MachXO5-NX JTAG port, refer to the JTAG Port Timing Specifications of MachXO5-NX Family Datasheet (FPGA-DS-02102).

6.4.3. JTAG ispTracy/Reveal Support

The JTAG ispTracy feature supported in FPGAs is the optional addition of internal logic analyzers (ILA). These ILAs have features similar 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 6.20.

To be IEEE 1149.1 compliant, there is a default 16-bit register implemented for registers ER1 and ER2 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 implemented in the 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 IEEE 1149.1 compliance, the BSDL file needs to be modified by the customer to define the lengths of these new ER1 and/or ER2 registers. This new implementation is different from that of all previous device architectures.

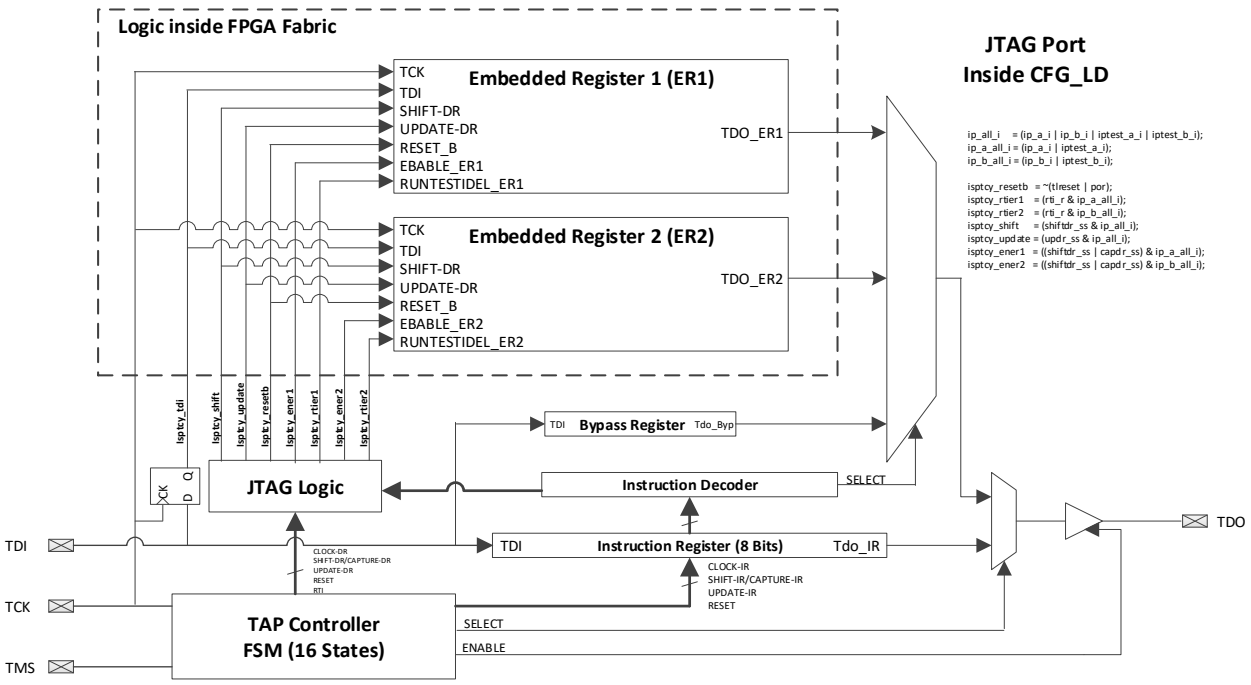


Figure 6.20. JTAG ispTracy Interface

In user functional mode, the four JTAG pins located in I/O Bank 1 can have their functionality changed from user function to JTAG function by bringing JTAG_EN 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 MachXO5-NX needs to match the JTAG pins I/O type. For example, if four JTAG pins in Bank 1 use LVCMOS33 then the JTAG driver needs to be 3.3 V.

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

After the JTAG_EN 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 JTAG_EN goes low, they remain 8 mA drive strength until switching.

6.5. Device Status Register

The MachXO5-NX 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^2C port, Slave I3C port, or LMMI interface.

Table 6.10. 64-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; Program input data to EFUSE Memory directly, Read back data from Shadow register.
					010 EFUSE Pseudo; Program input data to Shadow Register, Read back data from Shadow register.
					011 EFUSE Safe; Program Shadow Register data into EFUSE memory, Read back data from Shadow 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: • Selected configuration target security bit is set • Password protection is enabled and password is mismatched.
11	Read Enable	0	Not Readable	Readable	Read-protected from at least one of the following setup: • Security bit is set • Password protection is enabled and password is mismatched.
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

Bit	Function	Status Value			Comments
		Reset Value	0	1	
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
[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 time out when trying to read the signature from flash
43	Key Destroy Pass	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

Bit	Function	Status Value			Comments
		Reset Value	0	1	
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. 00 – No Auth 01 – ECDSA 10 – HMAC 11 – No Auth
50	Authentication Done	0	No	Yes	Indicates that authentication has completed.
51	Dry Run Authentication Done	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^2C/I3C Locked	0	No	Yes	Indicates that the I^2C/I3C port is locked
55	PUB Read Lock	0	No	Yes	ECDSA Public key read lock is enabled.
56	PUB Write Lock	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

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 6.11.

Table 6.11. 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 go low during reconfiguration. When this bit is set to *1*, the wake-up signals GOE, GWE and GSRN are set according to the user setting of NDR at the time of 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 tri-stated 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.

BIT	Description	Default	Note
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); 1 – Use boot address from CIB SRAM bits depend on bitstream setting. 0 – Use hard coded boot address (H000000 for first boot; HFFFF00 for second boot)
[18:17]	P_DONE CRTL	0	Overload control for PROGRAM_DONE command in bitstream for configuration daisy chain setup: 10 – Overload with BYPASS 11 – Overload with FLOW_THROUGH 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: 00 – Slow 01 – Medium 10 – Fast 11 – Fast
[5:0]	On-chip Flash Clock Select	0	Control bits that set the divide ratio to derive the 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. You 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 6.12.

Table 6.12. 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. 0 – Sampling of data occurs at the rising edge of the clock 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 0 – Active high clocks selected. In idle state clock is low 1 – Active low clocks selected. In idle state clock is high
[21:18]	Reserved	0	—
17	32-bit SPIM address	0	Enable 32-bit flash addressing mode, default 24-bit addressing. Fixed to be 0.
16	Reserved	0	—
15	Reserved	0	—
14	32-bit SPIM commands	0	Send 32-bit command set on flash, default 24-bit command set. Fixed to be 0.
13	Disable I/O glitch filter	0	Disable I/O Glitch Filter during configuration
12	Reserved	0	—
[11:8]	Reserved	0000	—

Bit	Definition	Default	Note
[7:5]	Reserved	0	—
4	Reserved	0	—
[3:2]	Reserved	0	—
1	Reserved	0	—
0	I/O Ready Disable	0	When set to 1, it disables waiting for I/O ready for device wakeup.

6.8. TransFR Operation

The MachXO5-NX device provides the TransFR™ capability. TransFR is described in Minimizing System Interruption During Configuration Using TransFR Technology (FPGA-TN-02198).

6.9. SPI/I²C/I3C Command Support

For the ISC and programming commands for all non-JTAG configuration interfaces such as SPI, I²C, and I3C, refer to the sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099).

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 6.13. JTAG Instruction Table

JTAG Instruction	OPCODE		Source Data at Capture-DR	Target Data at Update-DR	Target Data at Runtest Idle	Shift 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.
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	Read the 64-bit UNIQUE ID, 56 bits set by Lattice, 8 bits set by customer.

JTAG Instruction	OPCODE		Source Data at Capture-DR	Target Data at Update-DR	Target Data at Runtest Idle	Shift Register	Note
	Binary	Hex					
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. Bit 0: Cause a Global Set/Reset to occur on the device. Bit 1: Launch a One-time check of SED. Bit 2: Reserved Bit 3: Cause a reset on configuration logic related FSM and flags. Bit 4: Reserved for internal dry-run. Bit [6:5]: Launch Dry-Run event. 2'b01 – goes to BOOT 0 only 2'b10 – goes to BOOT 2 only 2'b11 – goes to BOOT 1; if failed, goes to BOOT 2 2'b00 - no dry-run 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/O are tri-stated 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 enter 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

JTAG Instruction	OPCODE		Source Data at Capture-DR	Target Data at Update-DR	Target Data at Runtest Idle	Shift Register	Note
	Binary	Hex					
							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 based on the 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

JTAG Instruction	OPCODE		Source Data at Capture-DR	Target Data at Update-DR	Target Data at Runtest Idle	Shift Register	Note
	Binary	Hex					
							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 of the address.
LSC_PROG_CTRL0	00100010	22	N/A	Control Register0	N/A	32-bit shift Register	Modify the Control Register 0.
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). See additional note[2].
LSC_READ_PASSWORD	11110010	F2	Password	N/A	N/A	128-bit shift Register	Read out the 128-bit password before activated for verification

JTAG Instruction	OPCODE		Source Data at Capture-DR	Target Data at Update-DR	Target Data at Runtest Idle	Shift Register	Note
	Binary	Hex					
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. See additional note[2].
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[1]	11100100	E4	N/A	N/A	N/A	96-bit shift register	Program security related feature such as Decrypt Enable, etc. based 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, and others based on the selection from the operand.
LSC_PROG_FEABITS[1]	11111000	F8	N/A	N/A	N/A	16-bit shift register	Program security related feature bits. See additional note[2].
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. See additional note[2].

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_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	Read back 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 Bit [1]=1; Release right I/O BANK (Bank 1 and Bank 2) Bit [0]=1; Release left I/O BANK (Bank 6 and Bank 7)
LSC_PROG_ECDSA_PUBKEY	01011001	59	N/A	N/A	N/A	512-bit shift register	Program the ECDSA Public Key. See additional note[2].

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_ECDSA_PUBKEY	01011010	5A	N/A	N/A	N/A	512-bit shift register	Read Back the ECDSA Public Key
LSC_READ_DR_UES	01011101	5D	Dry Run UES	N/A	N/A	32-bit shift Register	Read the dry-run User Electronic Signature shadow register.

Notes:

1. For the data field definition, refer to Table 6.23 and Table 6.24 in sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099).
2. JTAG instruction operation programs the internal EFUSE memory (non-volatile memory), which is one-time programmable (OTP).

7. Flash Programming

The MachXO5-NX's internal flash memory is the heart of the FPGA's configuration system. It is flexible, allowing you to store the FPGA's configuration data, as well as storing design specific data in the internal memory. It is also a resource that uses a precise erase and programming sequence. Lattice provides several methods for programming the MachXO5-NX device flash memory:

- JTAG or Slave SPI programming
- VMEmbedded: 'C' source for use with an embedded microprocessor controlling the JTAG port
- SSPIEmbedded: 'C' source for use with an embedded microprocessor controlling the SSPI port
- Custom: Permits creation of a custom solution.

The flash memory space can be accessed by the JTAG and SPI ports. These configuration ports may use offline or transparent modes to erase, program, and verify the MachXO5-NX flash memory resources. The LMMI interface is only permitted to use transparent programming operations. The sequence and timing of the commands presented to the Configuration Logic are identical across all the configuration ports. There are slight differences due to communication protocol standards when transmitting commands and data.

7.1. Flash Memory Space Partition

Each MachXO5-NX device contains a certain quantity of internal memory. The amount of memory depends on the device density of the MachXO5-NX device. Table 7.1 and Table 7.2 show the number of internal memory blocks available for the LFMXO5-25 and LFMXO5-55T/100T devices respectively. Each block represents 64k bytes of data.

Table 7.1. Number of Blocks of Flash Memory for the LFMXO5-25

Blocks (64 kB)	Address	Function	Default Usage	
1	0x00_0000	Header and Jump Table	MCS File	1
11	0x01_0000	CFG0	IMG0	11
4	0x0C_0000	UFM0 (EBR and LRAM INIT)	UFM0	4
11	0x10_0000	CFG1	IMG1	11
4	0x1B_0000	UFM1 (EBR and LRAM INIT)	UFM1	4
11	0x1F_0000	CFG2	IMG2	11
4	0x2A_0000	UFM2 (EBR and LRAM INIT)	UFM2	4
18	0x2E_0000	UFM (Up to 9)	User Data 0	2
	0x30_0000		User Data 1	2
	0x32_0000		User Data 2	2
	0x34_0000		User Data 3	2
	0x36_0000		User Data 4	2
	0x38_0000		User Data 5	2
	0x3A_0000		User Data 6	2
	0x3C_0000		User Data 7	2
	0x3E_0000		User Data 8[1]	1
1	0x3F_0000	Backup Jump Command[2]	MCS File	1

Notes:

1. User Data 8 has two blocks in the Radiant Programmer 2023.1 and older version. From software version 2023.2 onwards, the last block is allocated for the Backup Jump Command.
2. This option is available from the Radiant Programmer version 2023.2.

Table 7.2. Number of Blocks of Flash Memory for the LFMXO5-55T/100T

Blocks (64 kB)	Address	Function	Default Usage	
1	0x00_0000	Header and Jump Table	MCS File	1
33	0x01_0000	CFG0	IMG0	33
15	0x22_0000	UFM0 (EBR and LRAM INIT)	UFM0	15
33	0x31_0000	CFG1	IMG1	33
15	0x52_0000	UFM1 (EBR and LRAM INIT)	UFM1	15
33	0x61_0000	CFG2	IMG2	33
15	0x82_0000	UFM2 (EBR and LRAM INIT)	UFM2	15
110	0x91_0000	UFM (Up to 9)	User Data 0	13
	0x9E_0000		User Data 1	13
	0xAB_0000		User Data 2	13
	0xB8_0000		User Data 3	13
	0xC5_0000		User Data 4	13
	0xD2_0000		User Data 5	13
	0xDF_0000		User Data 6	13
	0xEC_0000		User Data 7	13
	0xF9_0000		User Data 8[1]	6
1	0xFF_0000	Backup Jump Command[2]	MCS File	1

Notes:

1. User Data 8 has seven blocks in the Radiant Programmer 2023.1 and older version. From software version 2023.2 onwards, the last block is allocated for the Backup Jump Command.
2. This option is available from the Radiant Programmer version 2023.2.

7.2. Flash Access Through Configuration Port

The Programmer tool provides the on-chip flash programming through the JTAG and Slave SPI configuration port. For the access mode, it provides both the offline mode and background (Transparent) mode.

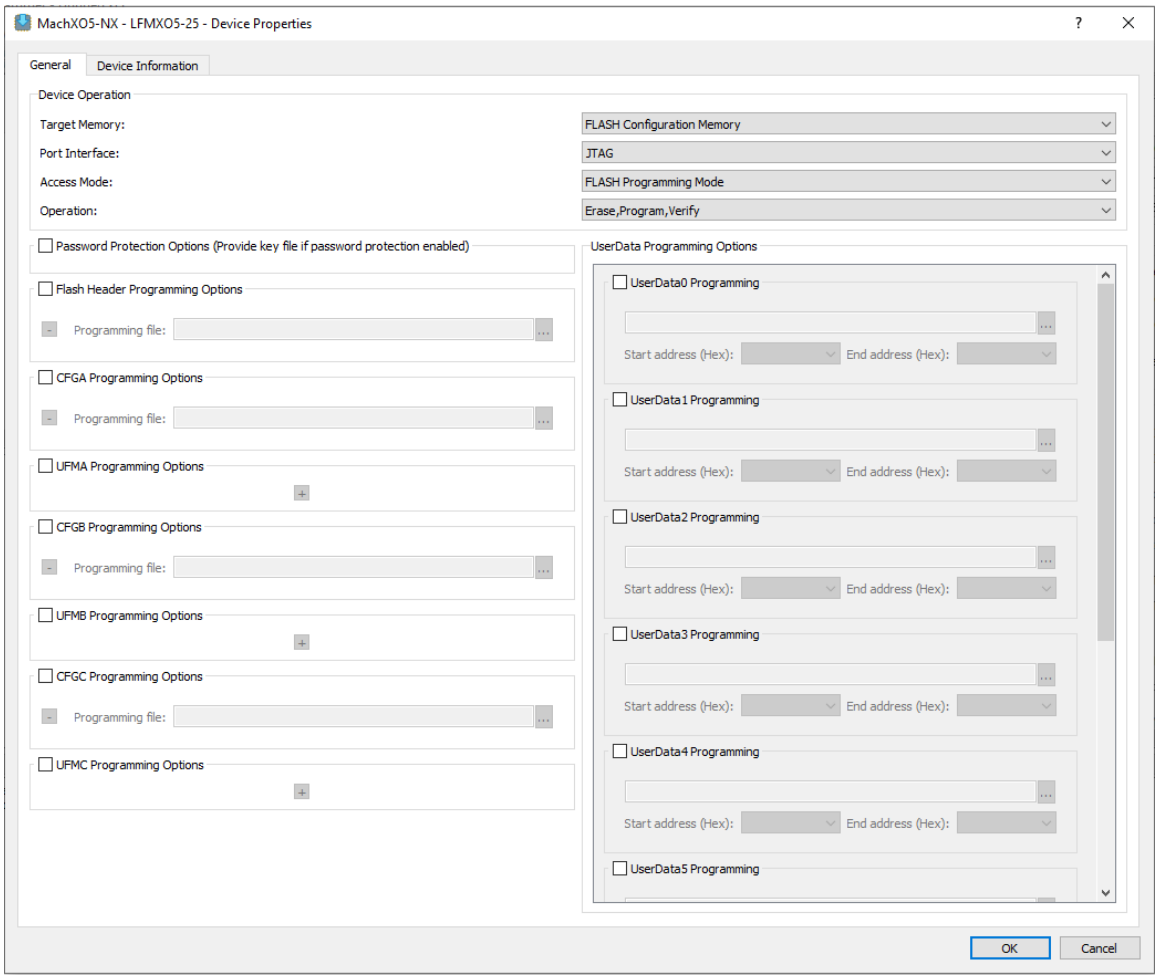


Figure 7.1. Radiant Programmer Operation for Flash Programming

7.3. Flash Programming Flow

The Radiant software flash programming process contains the following main steps with associated commands:

- INIT: Initializing.
- READ_ID: Command 0xE0 to check the MachXO5-NX IDCODE.
- FLASH_INITIAL: Shift in BYPASS(0xFF) opcode, then shift SPI_PROGRAM (0x3A) opcode to enable the SPI access to the on-chip flash.
- FLASH_CHECK_ID: Shift in IDCODE(0xAB) opcode to read back the on-chip flash ID.
- CFG_PROGRAM_VERIFY: Shift in Block Erase(0xDB) opcode to erase the CFG partitions, shift in Page Program(0x02) opcode to write the data.
- UFM_PROGRAM_VERIFY: Shift in Block Erase(0xDB) opcode to erase the UFM(A/B/C and 0-8) partitions, shift in Page Program(0x02) opcode to write the data.
- FLASH_FINALIZE: Shift in BYPASS(0xFF) opcode, then disable the SPI interface.

Table 7.3. Flash Erase/Program/Verify Flow

Instructions	Shift In Command	Shift In Data	Expected Output
Initialization			
Check MachXO5-NX ID	0xE0	—	0x010F7043
Bypass	0xFF	—	—
Enable SPI Program	0x3A	—	—
Check Flash ID	0xAB	—	0x15
Block Erase 0	0xD8	0x010000	—
Block Erase 1	0xD8	0x020000	—
…….	…….	…….	…….
Block Erase 10	0xD8	0x0B0000	—
Page Program 0	0x02	0x010000 + 256 Byte Data	—
Page Program 1	0x02	0x010100 + 256 Byte Data	—
…….	…….	…….	…….
Page Program 2815	0x02	0x0BFF00 + 256 Byte Data	—
Page Read for Verify 0	0x0B	—	256 Byte Data
Page Read for Verify 1	0x0B	—	256 Byte Data
…….	…….	…….	…….
Page Read for Verify 2815	0x0B	—	256 Byte Data
Finalization	0xFF	—	—

7.4. Flash Programming Commands

For the on-chip flash programming, the MachXO5-NX configuration engine uses the standard SPI flash programming command for the erase/program/verify operations. Table 7.4 lists the instructions used.

Table 7.4. Instruction Table

Instructions	Byte1	Byte2	Byte3	Byte4	Byte5	Byte6	Byte7
Write Enable	06h	—	—	—	—	—	—
Write Disable	04h	—	—	—	—	—	—
Release Power-Down /ID	ABh	Dummy	Dummy	Dummy	ID(7:0)	—	—
Read Data	03h	A23-A16	A15-A8	A7-A0	(D7-D0)	—	—
Fast Read	0Bh	A23-A16	A15-A8	A7-A0	Dummy	(D7-D0)	—
Page Program	02h	A23-A16	A15-A8	A7-A0	(D7-D0)	—	—
Block Erase (64KB)	D8h	A23-A16	A15-A8	A7-A0	—	—	—
Chip Erase	C7h/60h	—	—	—	—	—	—
Power-Down	B9h	—	—	—	—	—	—

7.5. Flash Access through LMMI Bus

The Flash Access soft IP enables you to perform write and read access to the internal flash memory of MachXO5-NX device. The write and read access is performed through the LMMI interface.

When a flash access operation is on-going, the TDO line of the hard JTAG port is disconnected. Loading the Reveal analyzer/controller and programming/configuration cannot be done using the hard JTAG port when there is an on-going flash access operation. Connection to the TDO line is restored after the flash access operation completes.

For detail information regarding user mode flash access operation, refer to the Flash Access IP Core (FPGA-IPUG-02171).

7.6. Dual-Boot and Multi-Boot Configuration

Both the primary and the golden (fail-safe) configuration data is stored in on-chip flash 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 on-chip memory

A CRC error is caused by incorrect data being written into the SRAM configuration memory. Data is read from the on-chip 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 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 MachXO5-NX device counts the number of 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 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 on-chip flash 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 software implementations.

The MachXO5-NX device supports dual-boot with the self-download mode. If the primary pattern fails to load correctly, the MachXO5-NX device starts loading data from the golden sector in the on-chip flash memory. A blank on-chip flash memory 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 users to split an on-chip flash memory 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 (0x3FFF00). This allows the system to boot to a known operable state, so that it continues to operate if for some reason (such as a power failure) the on-chip flash memory fails to program correctly.

Users can dynamically switch between up to two different design revisions stored in on-chip 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 one alternate bitstream.

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 through any sysCONFIG port causes the device to load from the defined on-chip Flash address. Dual-boot can also be deployed with multi-boot, allowing a golden (fail-safe) design (or third design) to be available in the on-chip flash memory. For more information regarding multi-boot operation, refer to the Multi-Boot Usage Guide for Nexus Platform (FPGA-TN-02145).

The Lattice Radiant Deployment Tool allows you to assemble flash memory images formatted to correctly match the hardware sector mapping.

7.7. Ping-Pong Boot

The Ping-Pong boot mode uses the jump table to select an image for booting without changing the location of the image in the flash memory. This is done with a jump table starting at address 0x000000 in the flash memory containing the following two instructions:

- LSC_PROG_SEC_BOOT (0x7F) – Specifies the secondary image boot address.
- JUMP (0x7E) – Specifies the primary image boot address. This command works only when it is specified in the bitstream, JTAG and slave ports do not support this command.

In case the jump table is corrupted by a power failure, a backup JUMP command is programmed at the last 256 bytes of the internal flash, that is the offset 0x3FFF00 for LFMXO5-25 devices or 0xFFFF00 for LFMXO5-55T/100T devices. The backup JUMP command is pointing to the secondary image by default. Refer to Figure 7.2 for the two images stored in the flash bitstream 0 and bitstream 1, where CFGx can be CFG0, CFG1, or CFG2. You may refer to Table 7.1and Table 7.2 for the respective CFG0/1/2 address offset.

Note that when configuration engine attempts to read the backup JUMP command, it always issue a read offset address at 0xFFFF00, the on-chip flash truncates the most significant bit of the address depending on the density of the flash. For example, the 32 Mb on-chip flash in the LFMXO5-25 device receives 0x3FFF00 even though the configuration engine sends 0xFFFF00.

To swap the order of booting, the jump table at the address of 0x000000 needs to be re-programmed. It is also optional to re-program the backup JUMP command at the offset 0x3FFF00/0xFFFF00 to point to the new secondary image.

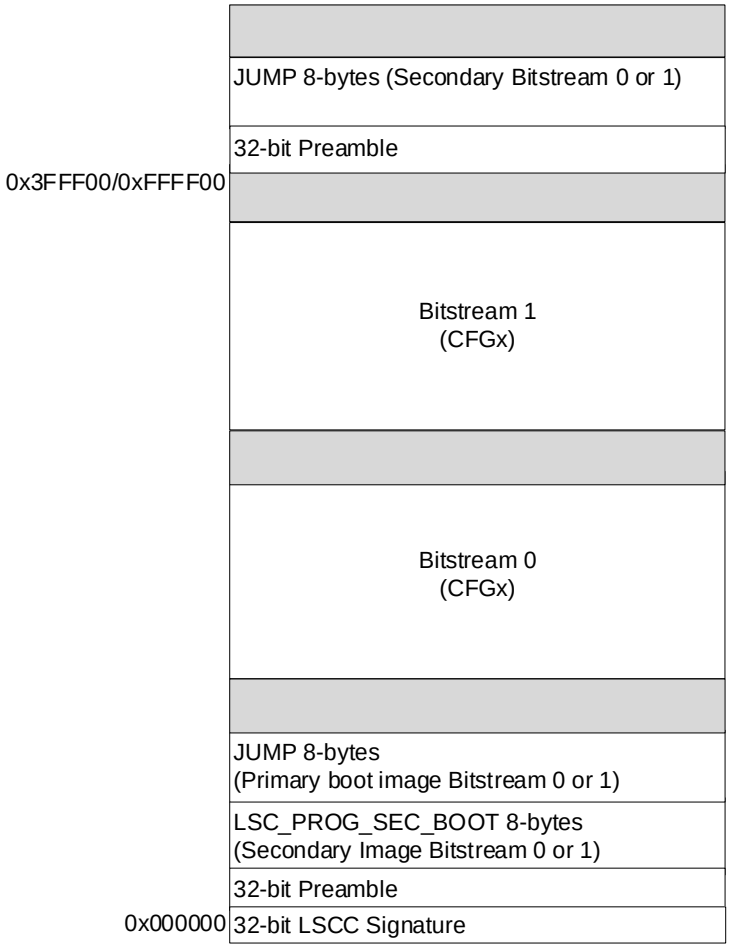


Figure 7.2. Jump Table Format

The default programming option in the Radiant Programmer to program on-chip flash memory is corresponding to the Ping-Pong Boot. The jump table is the Flash Header file in Intel Hex format with the .mcs file extension, and the bitstream 0 and 1 are corresponding to CFG0 and CFG1 respectively.

Important note: If the bitstream corruption happen at the 32-bit preamble of the primary image, the configuration engine does not load the secondary image correctly, both the BSE Error Code and BSE Error 1 Code in Device Status Register shows a preamble error. You can recover it by reprogramming the on-chip flash memory with a good image through the JTAG or SSPI port. Lattice recommends that you use Dual-Boot configuration if your application requires you to update the user image remotely.

7.8. Enabling Fast Configuration

In the MachXO5-NX device Radiant Project, the Radiant software generates the following files after a full design compilation:

- .bit file – to configure the SRAM of the MachXO5-NX device through the JTAG Fast Configuration. The configuration is volatile.
- .jed file – the user image itself, which carries the flash clock frequency setting that you set in the Radiant Device Constraint Editor. You can program the .jed file to the CFG0 or CFG1 slot of the on-chip flash memory.
- .mcs file – the header file or jump table, which contains the JUMP command that points to the primary image, and LSC_PROG_SEC_BOOT instruction that points to the secondary image. The JUMP operand1 is the same as the LSC_REFRESH, which specifies the flash read mode. For example, normal read, FAST serial read, DUAL read, and QUAD read.

The Radiant software is using the Ping-Pong Boot approach by generating the header file or jump table in the .mcs file format, and the user image in the .jed format. By default, the Radiant software generates the header file with normal read option. To minimize the boot time, you can set higher flash clock frequency in the Radiant Device Constraint Editor and use the Radiant Deployment Tool to generate the header file in QUAD read mode. Follow the instructions in the following sections to set the faster flash clock frequency in your Radiant software project, and to generate the header file with QUAD read operation.

7.8.1. Generating User Image with Faster Flash Clock Frequency

In your Radiant software project, under the Device Constraint Editor's Global tab, set the flash clock frequency to 112.5 MHz and the configuration I/O slew rate to fast.

- FLASH_CLK_FREQ = 112.5
- CONFIG_IOSLEW = FAST

The Radiant software generates the .jed file after the compilation. This file carries the flash clock frequency and configuration I/O slew rate setting.

7.8.2. Generating Header File with QUAD Read Mode

To generate the header file with QUAD Read mode:

3. Launch the Radiant Deployment Tool[(1)] from the Radiant Programmer Tools option. In the **Function Type** field, select *External Memory* and in the **Output File Type** field, select *Ping-Pong Boot,* as shown in Figure 7.3. Click **OK** to proceed.
   Note: The Deployment Tool 2023.1 has a known issue to generate a header file with Quad read mode. Lattice recommends that you use version 2022.1SP1 or later versions.

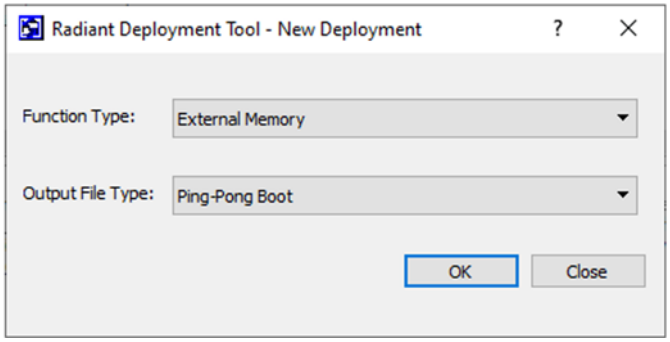


Figure 7.3. New Deployment Option in the Radiant Deployment Tool

4. Select two MachXO5-NX bitstream files (.jed/.bit/.rbt). You can select the same bitstream file since the objective is to generate the header file only, as shown in Figure 7.4. Click **Next** to proceed.

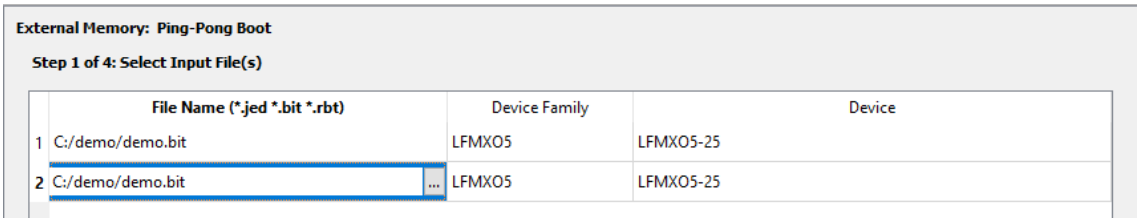


Figure 7.4. Select Input File for Ping-Pong Boot

5. At the **Options** tap, set the following settings, and click **Next** to proceed (see Figure 7.5):
   - Output Format: Intel Hex
   - SPI Flash Size (Mb): 32 (LFMXO5-25) or 128 (LFMXO5-55T/100T)
   - SPI Flash Read Mode: Quad I/O SPI Flash Read
   - Primary Pattern: (default), Starting Address: 0x00010000 (CFG0), or 0x00100000 (CFG1) (as applicable)
   - Secondary Pattern: (default), Starting Address: 0x00010000 (CFG0), or 0x00100000 (CFG1) (as applicable)
   - First Boot Selection: Primary Pattern or Secondary Pattern (as applicable)
   - Check the “Generate Jump Table Only” checkbox – this generates the header file only to be used later in the Radiant Programmer.

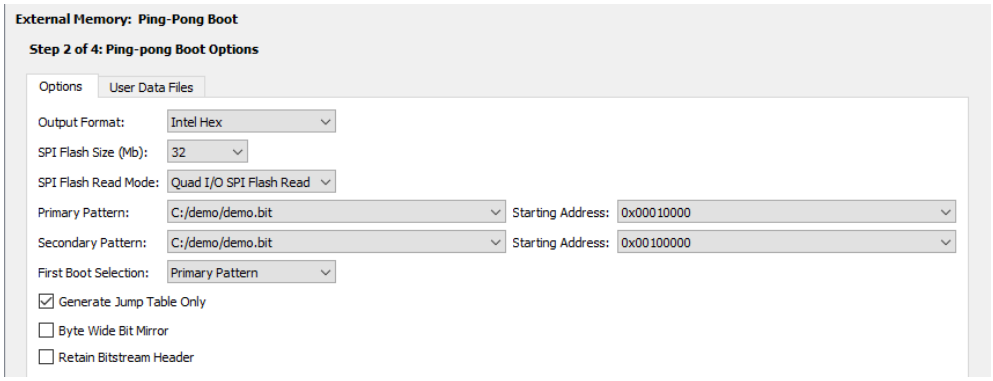


Figure 7.5. Ping-Pong Boot Options

6. Set the .mcs output file name and click **Next** to proceed.
7. Verify the summary and click **Generate** to generate the header file.

7.8.3. Programming Header File and User Image Using the Radiant Programmer

You can launch the standalone Radiant Programmer or the Programmer from the Radiant Tools menu and configure the setting as shown in Figure 7.6 to program the header file and user image to the on-chip flash memory.

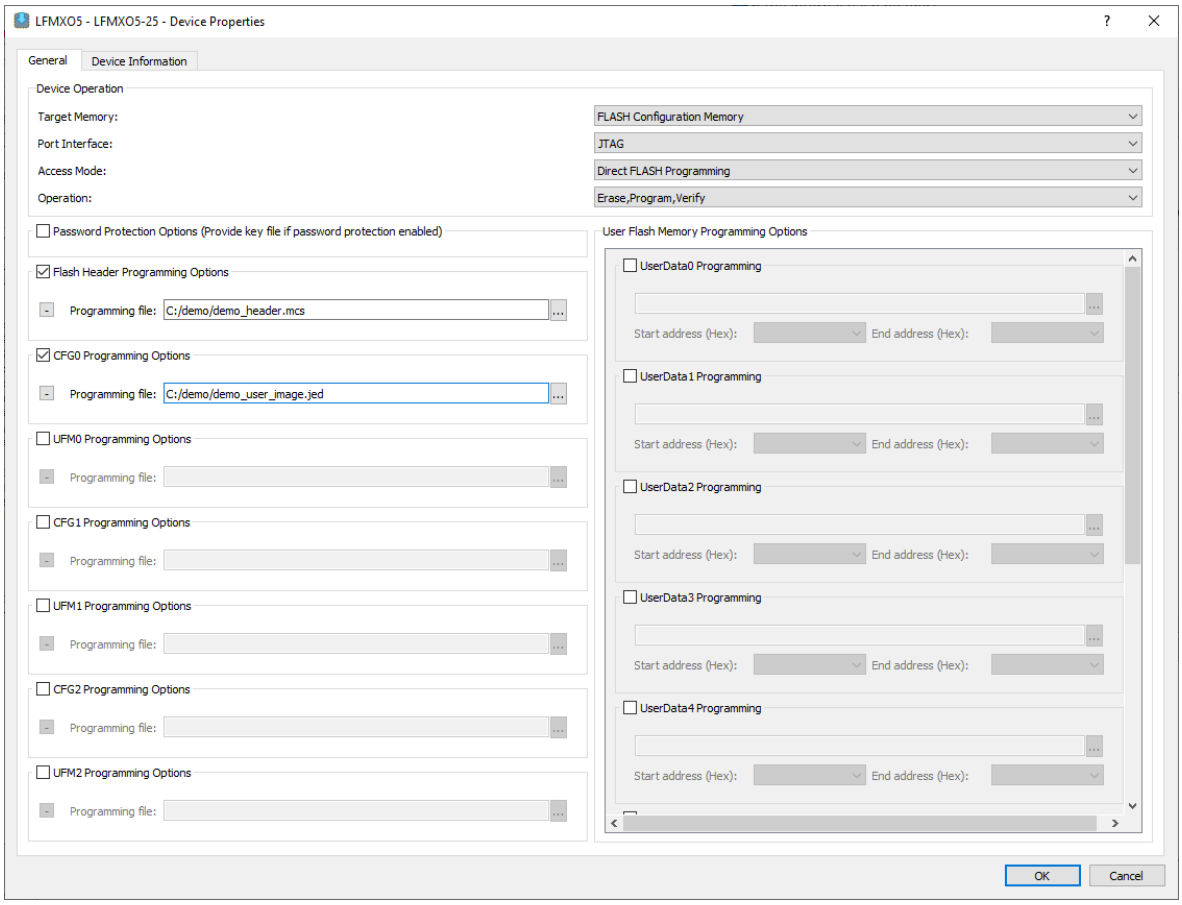


Figure 7.6. Programmer Options to Program Header File and User Image

From the Radiant Programmer version 2023.2, when you program the header file and user image, the programmer will program the backup JUMP command automatically at the last 256-byte of the on-chip flash memory.

Note that these instructions are applicable to Ping-Pong Boot, which relies on the jump table to boot the primary image. For Dual-Boot use case, refer to the Multi-Boot Usage Guide for Nexus Platform (FPGA-TN-02145) for similar guidelines.

8. Software Selectable Options

The operation of the MachXO5-NX device configuration logic is managed by options selected in the Lattice Radiant design software. The MachXO5-NX 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 8.1.

▾ SysConfig	
SLAVE_SPI_PORT	DISABLE
SLAVE_I2C_PORT	DISABLE
SLAVE_I3C_PORT	DISABLE
JTAG_PORT	ENABLE
DONE_PORT	ENABLE
INITN_PORT	ENABLE
PROGRAMN_PORT	ENABLE
BACKGROUND_RECONFIG	OFF
DONE_EX	OFF
DONE_OD	ON
FLASH_CLK_FREQ	3.5
TRANSFR	OFF
CONFIG_IOSLEW	SLOW
CONFIG_SECURE	OFF
WAKE_UP	ENABLE_DONE_SYNC
COMPRESS_CONFIG	OFF
EARLY_IO_RELEASE	OFF
BOOTMODE	DUAL
CONFIGIO_VOLTAGE_BANK0	NOT_SPECIFIED
CONFIGIO_VOLTAGE_BANK1	NOT_SPECIFIED
FLASH_PREAMBLE_TIMER_CYCLES	600000
CONFIGURATION	CFGUFM
CUR_DESIGN_BOOT_LOCATION	IMAGE_0
PRIMARY_BOOT	IMAGE_0
SECONDARY_BOOT	NONE
▾ User Code	
UserCode Format	Binary
UserCode	00000000000000000000000000000000
Unique Id	0000

Figure 8.1. sysCONFIG Preferences in Global Tab, Lattice Radiant Device Constraint Editor

Table 8.1. sysCONFIG Options

Option Name	Default Setting	All Settings
SLAVE_SPI_PORT	DISABLE	DISABLE, SERIAL, DUAL, QUAD
SLAVE_I2C_PORT	DISABLE	DISABLE, ENABLE
SLAVE_I3C_PORT	DISABLE	DISABLE, ENABLE
JTAG_PORT	ENABLE	DISABLE, ENABLE
DONE_PORT	ENABLE	DISABLE, ENABLE
INITN_PORT	ENABLE	DISABLE, ENABLE
PROGRAMN_PORT	ENABLE	DISABLE, ENABLE
BACKGROUND_RECONFIG	OFF	OFF, ON, SRAM_EBR, SRAM_ONLY
DONE_EX	OFF	OFF, ON
DONE_OD	ON	OFF, ON
FLASH_CLK_FREQ	3.5	3.5, 7.0, 14.1, 28.1, 56.2, 90, 112.5, 150
TRANSFR	OFF	OFF, ON
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
BOOTMODE	DUAL	DUAL, SINGLE, NONE
CONFIGIO_VOLTAGE_BANK1	NOT_SPECIFIED	NOT_SPECIFIED ,2.5, 1.2, 1.5, 1.8, 3.3
CONFIGIO_VOLTAGE_BANK2	NOT_SPECIFIED	NOT_SPECIFIED, 2.5, 1.2, 1.5, 1.8, 3.3
FLASH_PREAMBLE_TIMER_CYCLES	600000	600000, 400, 2000, 4000, 20000, 40000, 200000, 400000
CONFIGURATION	CFGUFM	CFGUFM, CFM_EBRUFM, External, CFG
CUR_DESIGN_BOOT_LOCATION	IMAGE_0	IMAGE_0, IMAGE_1, IMAGE_2, External
PRIMARY_BOOT	IMAGE_0	IMAGE_0, IMAGE_1, IMAGE_2, External
SECONDARY_BOOT	NONE	NONE, IMAGE_0, IMAGE_1, IMAGE_2, External
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.

Note: 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.

8.1. SLAVE_SPI_PORT

The SLAVE_SPI_PORT allows you to preserve the Slave SPI configuration port after the MachXO5-NX 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 MachXO5-NX 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 MachXO5-NX 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 you 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 MachXO5-NX 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 you from over-assigning I/O to those pins.

The default setting for this preference is DISABLE.

8.2. SLAVE_I2C_PORT/SLAVE_I3C_PORT

The SLAVE_I2C_PORT/SLAVE_I3C_PORT allows you to preserve the I^2C/I3C configuration port after the MachXO5-NX device enters user mode. There are two states to which SLAVE_I2C_PORT/SLAVE_I3C_PORT preference can be set:

- ENABLE – This setting preserves the I^2C/I3C port pins (SD2/SCL, SD3/SDA) when the MachXO5-NX device is in user mode. It allows you to read all the configuration data, except the EBR and the distributed RAM contents in user mode. The preference also prevents you from over-assigning I/O to those pins.
- DISABLE – This setting disconnects the I^2C/I3C port pins (SD2/SCL, SD3/SDA) from the configuration logic. It allows I^2C/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.

8.3. JTAG_PORT

The JTAG_PORT allows you to preserve the JTAG configuration port after the MachXO5-NX device enters user mode. There are two states to which JTAG_PORT preference can be set:

- ENABLE – This setting preserves the JTAG port pins (TCK, TMS, TDI, and TDO) when the MachXO5-NX device is in user mode. The preference also prevents you 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.

8.4. 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 preserves the DONE pins as the sysCONFIG pin. This preference setting also prevents you 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.

8.5. 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 preserves 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.

8.6. 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 preserves the PROGRAMN pins as the sysCONFIG pin. This preference setting also prevents you 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.

8.7. 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.

8.8. 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 is based on your selection—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 setting is OFF.

8.9. 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 MachXO5-NX device. The default setting is ON. This ensures the needs to make a conscious decision to drive the pin.

8.10. Flash CLK Frequency

The FLASH_CLK_FREQ preference allows you to alter the CLK frequency used to retrieve data from an on-chip flash memory when using self-download configuration modes. The MachXO5-NX device uses a nominal 3.5 MHz (±15%) clock frequency to begin retrieving data from the on-chip flash memory. The FLASH_CLK_FREQ value is stored in the incoming configuration data. The MachXO5-NX device reads a series of padding bits, a *start of data* word (0xBDB3) and a control register value. The control register contains the new FLASH_CLK_FREQ value. The MachXO5-NX device switches to the new clock frequency shortly after receiving the FLASH_CLK_FREQ value. The FLASH_CLK_FREQ has a range of possible frequencies available from 3.5 MHz up to 133 MHz.

8.11. TRANSFR

The TransFR function used by the MachXO5-NX device requires the configuration data loaded into the configuration SRAM, and any future configuration data file loaded into the on-chip Flash memory have the TRANSFR set to the ENABLE state. See the TransFR Operation section and Minimizing System Interruption During Configuration Using TransFR Technology (FPGA-TN-02198) for more information about using TransFR.

8.12. CONFIG_IOSLEW

The CONFIG_IOSLEW preference provides the option for different I/O slew rates for configuration related I/O, which makes the MachXO5-NX devices easily adaptable to different board environments. There are three options for the CONFIG_IOSLEW preference, SLOW, MEDIUM, and FAST. The default setting for the CONFIG_IOSLEW preference is SLOW.

8.13. CONFIG_SECURE

When the CONFIG_SECURE preference is set to ON, the read-back function of the SRAM memory is blocked. The device must be reprogrammed 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.

8.14. WAKE_UP

The WAKE_UP preference provides options for you to choose different wakeup sequences 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 must 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 MachXO5-NX device wake up sequence.

8.15. COMPRESS_CONFIG

The COMPRESS_CONFIG preference alters the way files are generated with compressed FPGA data frames. The default setting of COMPRESS_CONFIG is OFF.

8.16. EARLY_IO_RELEASE

The EARLY_IO_RELEASE preference enables early I/O release feature on I/O Bank 1, I/O Bank 2, I/O Bank 6, and I/O Bank 7. The default setting of EARLY_IO_RELEASE is OFF.

8.17. 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) fails, the secondary (Golden) booting image is automatically invoked to boot up the device. This setting is applicable to Dual-Boot, Multi-Boot, and Ping-Pong Boot configurations.
- SINGLE – Perform the single boot only. If the booting fails, the device goes to the unprogrammed mode directly.
- NONE – No Self-Download 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.

8.18. CONFIGIO_VOLTAGE_BANK1/CONFIGIO_VOLTAGE_BANK2

Because the sysCONFIG pins used for configuration may or may not be used in the design, users can declare the voltage interface for the sysCONFIG banks (Bank 1 and Bank 2). Setting this attribute informs the software which voltage is required in this bank to satisfy the sysCONFIG requirements. DRC errors can then be generated based on CONFIGIO_VOLTAGE_BANK1/CONFIGIO_VOLTAGE_BANK2 and usage of the dual-purpose sysCONFIG pins. The default is NOT_SPECIFIED.

CONFIGIO_VOLTAGE_BANK1/CONFIGIO_VOLTAGE_BANK2 = NOT_SPECIFIED/1.2V/1.5V/1.8V/2.5V/3.3V (NOT_SPECIFIED = Default)

8.19. CONFIGURATION

The CONFIGURATION preference allows you to control the configuration memory sectors to store the configuration image. The CONFIGURATION preference has four possible settings:

- CFG –The configuration bitstream is stored in the CFG0, CFG1, or CFG2 flash sector. The configuration data includes EBR initialization data.
- CFG_EBRUFM – This preference creates configuration data that is stored in the CFG0, CFG1, or CFG2 flash sector. The EBR initialization data is stored in the lowest page addresses of the UFM0, UFM1, or UFM2 sector. The UFM0, UFM1, or UFM2 sector is available in user mode. You must restore the EBR initialization data when making changes to the UFM0, UFM1, or UFM2 to guarantee correct operation.
- CFGUFM – The CFGUFM preference is the default mode for building configuration data. This preference creates configuration data that is stored in the CFG0, CFG1, or CFG2 flash sector. This mode differs from CFG by allowing the configuration data to overflow into the UFM0, UFM1, or UFM2 sector. The configuration data increases in size as EBR initialization data is added to the design.

The CONFIGURATION preference defaults to the CFGUFM state in the current release of the Radiant software. The Radiant design software only generates bitstream files when the entire design fits within the flash memory.

8.20. CUR_DESIGN_BOOT_LOCATION

The CUR_DESIGN_BOOT_LOCATION preference is used to specify the location of the configuration image generated by the current Radiant software project in Ping-Pong Boot applications. When set to IMAGE_0, it is targeted to the CFG0 space of the internal flash memory. When set to IMAGE_1, it is targeted to the CFG1 space of the internal flash memory. When set to IMAGE_2, it is targeted to the CFG2 space of the internal flash memory.

8.21. PRIMARY_BOOT

The PRIMARY_BOOT preference allows you to control which configuration memory sector is used for the single image configuration mode or the first boot in the Dual Boot Configuration Mode. The PRIMARY_BOOT preference has three possible settings:

- IMAGE_0 – This preference is the default setting. It uses CFG0 or CFG0 plus UFM0 sectors for the primary boot. The CONFIGURATION preference determines whether UFM0 is used to store the configuration image.
- IMAGE_1 – This preference uses CFG1 or CFG1 plus UFM1 sectors for the primary boot. The CONFIGURATION preference determines whether UFM1 is used to store the configuration image.
- IMAGE_2 – This preference uses CFG2 or CFG2 plus UFM2 sectors for the primary boot. The CONFIGURATION preference determines whether UFM2 is used to store the configuration image.

8.22. SECONDARY_BOOT

The SECONDRY_BOOT preference allows you to control which configuration memory sector is used for the secondary boot in the Dual Boot Configuration Mode. The SECONDARY_BOOT preference has four possible settings:

- NONE – This preference is the default setting. It is used to select the single image configuration mode.
- IMAGE_0 – This preference is one of the settings for the golden image Dual Boot Configuration Mode. It uses CFG0 or CFG0 plus UFM0 sectors to store the golden configuration image for the secondary boot. The CONFIGURATION preference determines whether UFM0 is used to store the configuration image.
- IMAGE_1 – This preference is one of the settings for the golden image Dual Boot Configuration Mode. It uses CFG1 or CFG1 plus UFM1 sectors to store the golden configuration image for the secondary boot. The CONFIGURATION preference determines whether UFM1 is used to store the configuration image.
- IMAGE_2 – This preference is one of the settings for the golden image Dual Boot Configuration Mode. It uses CFG2 or CFG2 plus UFM2 sectors to store the golden configuration image for the secondary boot. The CONFIGURATION preference determines whether UFM2 is used to store the configuration image.

8.23. 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 four 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-bit is Unique ID and the lower 16 bits are sequentially increased automatically for every bitstream generation.

8.24. USERCODE

The MachXO5-NX 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. 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.

8.25. UNIQUE_ID

The MachXO5-NX 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.

9. Device Wake-up Sequence

When configuration is complete (the SRAM has been loaded), the device wakes up in a predictable fashion. If the MachXO5-NX 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 9.1 shows the wake-up sequence with DISABLE_DONE_SYNC option.

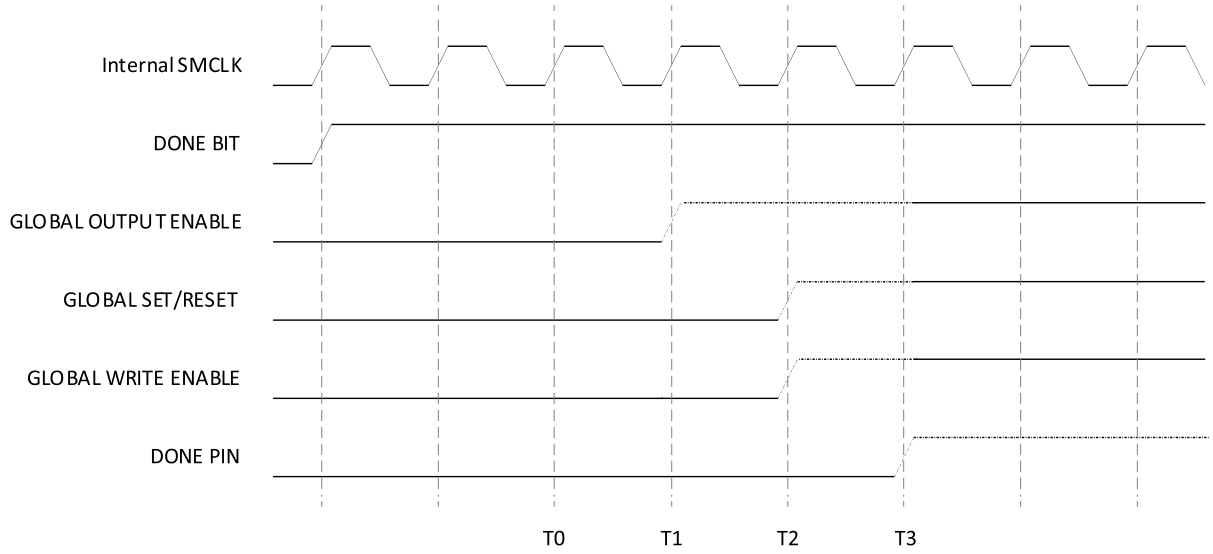


Figure 9.1. Wake-up Sequence Using Internal Clock

9.1. Wake-up Signals

Three internal signals GSRN, GWE, and GOE, determine the wake-up sequence.

- GSRN is used to set and reset the core of the device. GSRN is asserted (low) during configuration and de-asserted (high) in the wake-up sequence.
- When the GWE 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 MachXO5-NX 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 9.1. 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 9.1. Wake-up Sequences

Sequence	Phase T0	Phase T1	Phase T2	Phase T3
ENABLE_DONE_SYNC	DONE	GOE	GWE, GSRN	—
DISABLE_DONE_SYNC	—	GOE	GWE, GSRN	DONE

9.2. Wake-up Clock

The MachXO5-NX device always use the internal clock to perform the device wake up sequence. Once the MachXO5-NX device is configured, it enters the wake-up state, which is the transition from the configuration mode to user mode. This sequence is synchronized to the internal SMCLK, which is derived from internal oscillator.

10. 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 MachXO5-NX device supports only Flow Through chaining method with first FPGA being slave. The *Synchronous to External DONE pin* option (DONE_EX) must be enabled in Lattice Radiant Device Constraint Editor. Based on the configuration ports arrangement, the MachXO5-NX devices can only be the first device inside a configuration daisy chain.

10.1. Flow Through Option

The Flow Through option for the MachXO5-NX 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.

An example of the MachXO5-NX device in a configuration daisy chain with Flow Through option is shown in Figure 10.1.

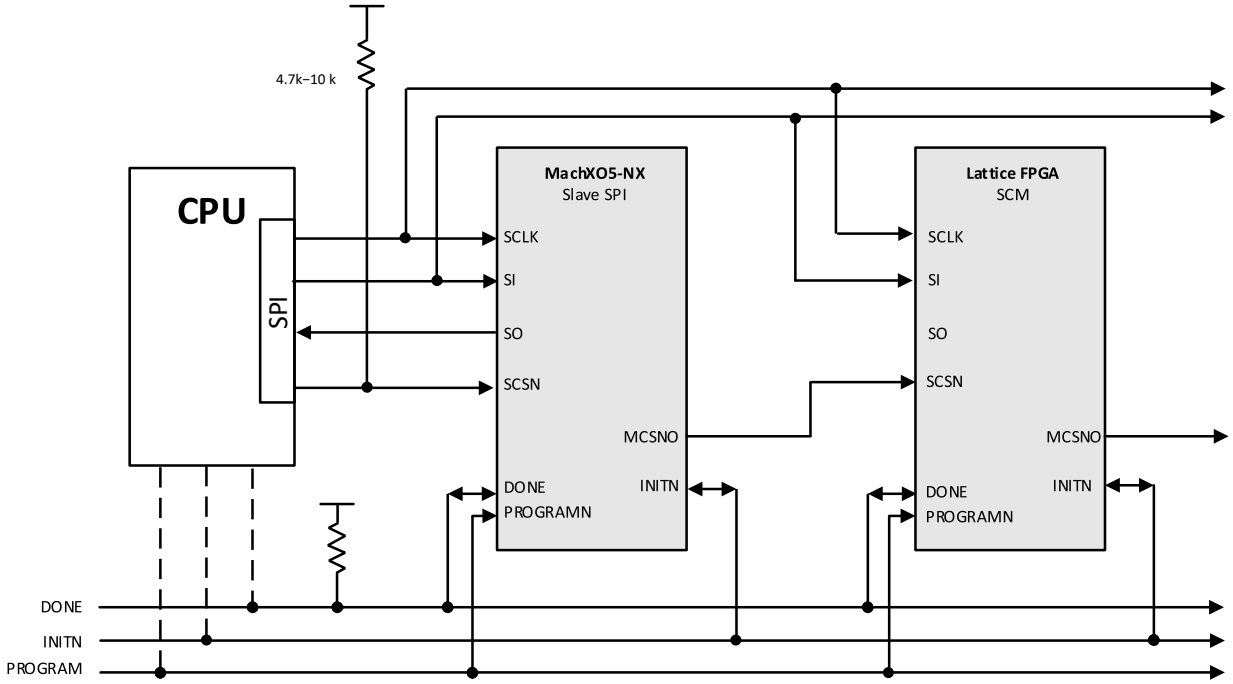


Figure 10.1. MachXO5-NX in Configuration Daisy Chain with Slave SPI in Flow Through Mode

Appendix A. MachXO5-NX Slave SPI Programming Guide

The SPI port of the MachXO5-NX device can be used for device configuration. If these pins are not used by user logic, they are tri-stated with a weak pull-up. The Slave SPI port must be enabled 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 through 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 users 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 -> SCLK
- 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 targets 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. MachXO5-NX 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, and ASCII) 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 must 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 must be padded with all 0 (3-bit 0s) at the rightmost to make it byte bounded (104-bit), 10……01000

Table B.1. MachXO5-NX 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) 1111111111111111	First 16 bits of the 32-bit Preamble	None
	1011110110110011	Second 16 bits of the 32-bit Preamble (0xFFFFBDB3)	
	11111111…11111111	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[3]	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[4]	Include

Frame	Contents[D7…D0]	Description	CRC
Frame (Store Compress)[8]	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 (Write Address for Loading MIB 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	Include
Frame (Auto Increment For next 32 Frames)	10111000	LSC_PROG_INCR_CMP Command.	Include
	1	CRC Comparison Flag (1 = yes)[10]	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
	0000000000100000	Number of MIB Frames Always 32 (16-bit Frame Size)	Include
Frame (Data 0)	(Frame 0 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	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)[6]	Include
	11111111…11111	Dummy byte (32 bits of ones)	Include
==========	==========	==========	==========
Frame (Data 31)	(Frame 31 Data)	*x* Data bits for Frame 31 (14 bits Parity + Data)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111…1111	Dummy byte (32 bits of ones)	Include: Start
Frame (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 (Write Increment)	10111000	LSC_PROG_INCR_CMP Command.	Include
	1	CRC Comparison Flag (1 = yes)[10]	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[6] For example, CrossLink-NX A. 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)[6]	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)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111…1111	Dummy byte (32 bits of ones)	Start: Include
Frame (Write address for Loading TAP 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	Include
Frame (Write Increment)	10111000	LSC_PROG_INCR_CMP Command.	Include
	1	CRC Comparison Flag (1 = yes)[10]	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[6]	Include
Frame (Data 0)	(Frame 0 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	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)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111…1111	Dummy byte (32 bits of ones)	Start: Include

Frame	Contents[D7…D0]	Description	CRC
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 (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[5] 0: 8-bit Hard IP 1: 10-bit Hard IP 2: 10-bit EBR /Large RAM 3: 32-bit Hard IP	Include
	Data [27:17]	ID CODE	Include
	Data[16:0]	Starting address	Include
Frame (Write data through Init Bus)	01110010	LSC_INIT_BUS_WRITE Command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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 (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)[10]	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 security option that users chose. It must be replaced with the LSC_PROG_SED_CRC frame by Bitgen if users 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 users 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. MachXO5-NX 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**) 1111111111111111	First 16 bits of the 32-bit Preamble	None
	1011110110110011	Second 16 bits of the 32-bit Preamble (0xFFFFBDB3)	
	11111111…11111111	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[3]	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[4]	Include
Frame (Write Address for Loading MIB 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[7]	Include
Frame (Auto Increment For next 32 Frames)	10000010	LSC_PROG_INCR command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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
	0000000000100000	(MIB Frames size 32) (16-bit Frame Size)[6]	Include
Frame (Data 0)	(Frame 0 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
Frame	(Frame 1 Data)	*x* Data bits for Frame 1(14 bits Parity + Data)[6]	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: Include
==========	==========	==========	==========
Frame (Data 31)	(Frame 31 Data)	*x* Data bits for Frame 31 (14 bits Parity + Data)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
Frame (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 (Write Increment)	10000010	LSC_PROG_INCR command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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[6] For Example, CrossLink-NX A. 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)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
==========	==========	==========	==========
Frame (Data n-1)	(Frame n-1 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
Frame (Write address for Loading TAP 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[7]	Include

Frame	Contents[D7…D0]	Description	CRC
Frame (Write Increment)	10000010	LSC_PROG_INCR command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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[6]	Include
Frame (Data 0)	(Frame 0 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	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)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: 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 (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[5] 0: 8-bit Hard IP 1: 10-bit Hard IP 2: 10-bit EBR /Large RAM 3: 32-bit Hard IP	Include
	Data [27:17]	ID CODE	Include
	Data[16:0]	Starting address	Include
Frame (Write data through Init Bus)	01110010	LSC_INIT_BUS_WRITE Command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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 (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)[10]	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 security option that users chose. It must be replaced with the LSC_PROG_SED_CRC frame by Bitgen if users 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 users 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. MachXO5-NX 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**) 1111111111111111	First 16 bits of the 32-bit Preamble.	None
	1011110110110011	Second 16 bits of the 32-bit Preamble (0xFFFFBDB3).	
	11111111…11111111	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[3]	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[4]	Include
Frame (Write Address for Loading MIB 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[7]	Include
Frame (Auto Increment For next 32 Frames)	10000010	LSC_PROG_INCR command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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
	0000000000010000	Number of MIB Frames Always 32 (16-bit Frame Size[6,7]	Include

Frame	Contents[D7…D0]	Description	CRC
Frame (Data 0)	(Frame 0 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
==========	==========	==========	==========
Frame (Data 31)	(Frame 31 Data)	*x* Data bits for Frame 31(14 bits Parity + Data)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
Frame (Release I/O) OPTIONAL FRAME (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 (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 (Write Increment)	10000010	LSC_PROG_INCR command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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[6] For Example, CrossLink-NX A. 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)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: Include
==========	==========	==========	==========
Frame (Data n-1)	(Frame n-1 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: 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
	1000000000100000	16-bit address to load	Include
Frame (Write Increment)	10000010	LSC_PROG_INCR command	Include
	1	CRC Comparison Flag (1 = yes)[10]	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[6]	Include
Frame (Data 0)	(Frame 0 Data)	*x* Data bits for Frame 0 (14 bits Parity + Data)[6]	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)[6]	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)[6]	Include: End
	(CRC[15..0])	16-bit CRC	CRC
	11111111	Dummy byte (8 bits of ones)	Start: 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 (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[5] 0: 8-bit Hard IP 1: 10-bit Hard IP 2: 10-bit EBR /Large RAM 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)[10]	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)[10]	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 that users chose. It must be replaced with the LSC_PROG_SED_CRC frame by Bitgen if users 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 users 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. MachXO5-NX 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
Control Register 0	(CtlReg0[0..31])	Control Register 0 Data
Device Specific Padding	1111…1111	Terminator bytes (all 1s). See Table B.5 for number of bytes.
Frame 0	1111…1111	Dummy (Stop) bytes (all 1s). See Table B.5 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 Table B.5 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 Table B.5 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

MachXO5-NX Device Specific

Table B.5. MachXO5-NX 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)
LFMXO5-25	6622	6	676	14	16	8	720
LFMXO5-15D	6622	6	676	14	16	8	720
LFMXO5-55T	16822	0	882	14	16	8	920
LFMXO5-100T	16822	0	882	14	16	8	920
LFMXO5-55TD	16822	0	882	14	16	8	920

Table B.6. MachXO5-NX Device ID

Device Family	Device Name	32-bit IDCODE
MachXO5-NX	LFMXO5-25	0x010F7043
MachXO5-NX	LFMXO5-15D	0x110F7043[1]
MachXO5-NX	LFMXO5-55T	0x610F4043
MachXO5-NX	LFMXO5-100T	0x110F4043
MaxhXO5-NX	LFMXO5-55TD	0x710F4043[1]

Note:

1. For MachXO5-NX Root-of-Trust devices, reading the device ID requires a special API. For more information, refer to the embedded security and function block user guide for the respective devices.

Appendix C. Modifying the Nexus™ Feature Row

The Feature Row contains design-specific information for non-volatile device operations and behaviors. For example, you can specify the device's I^2C address.

Breakdown of Feature Row Bits

The Feature Row has a total of 144 bits:

- 32-bit NV CR1 bits
- 96-bit FEATURE bits
- 16-bit FEABITS

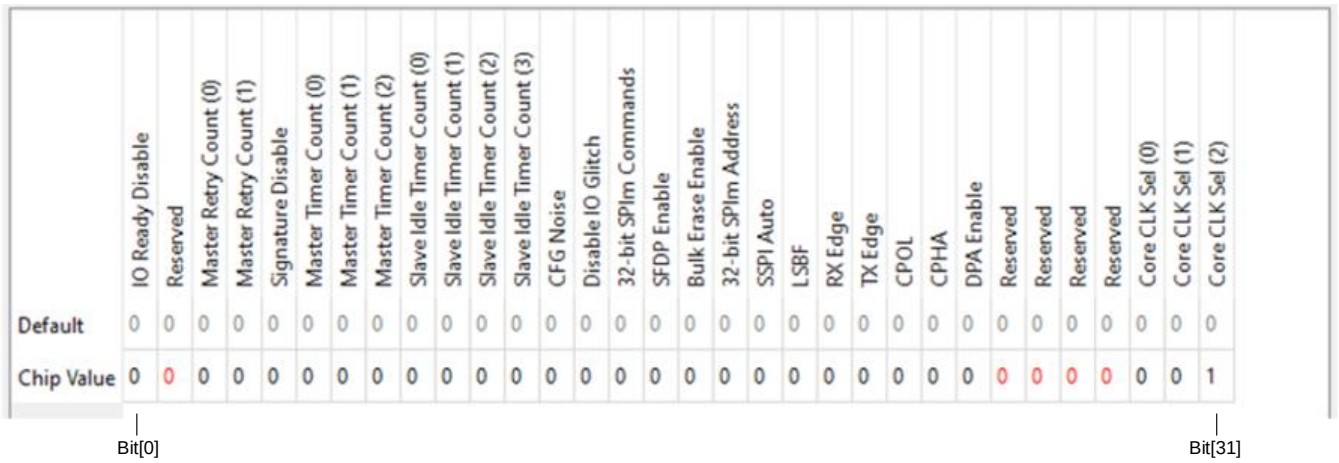


Figure C.1. 32-bit NV CR1 Bits

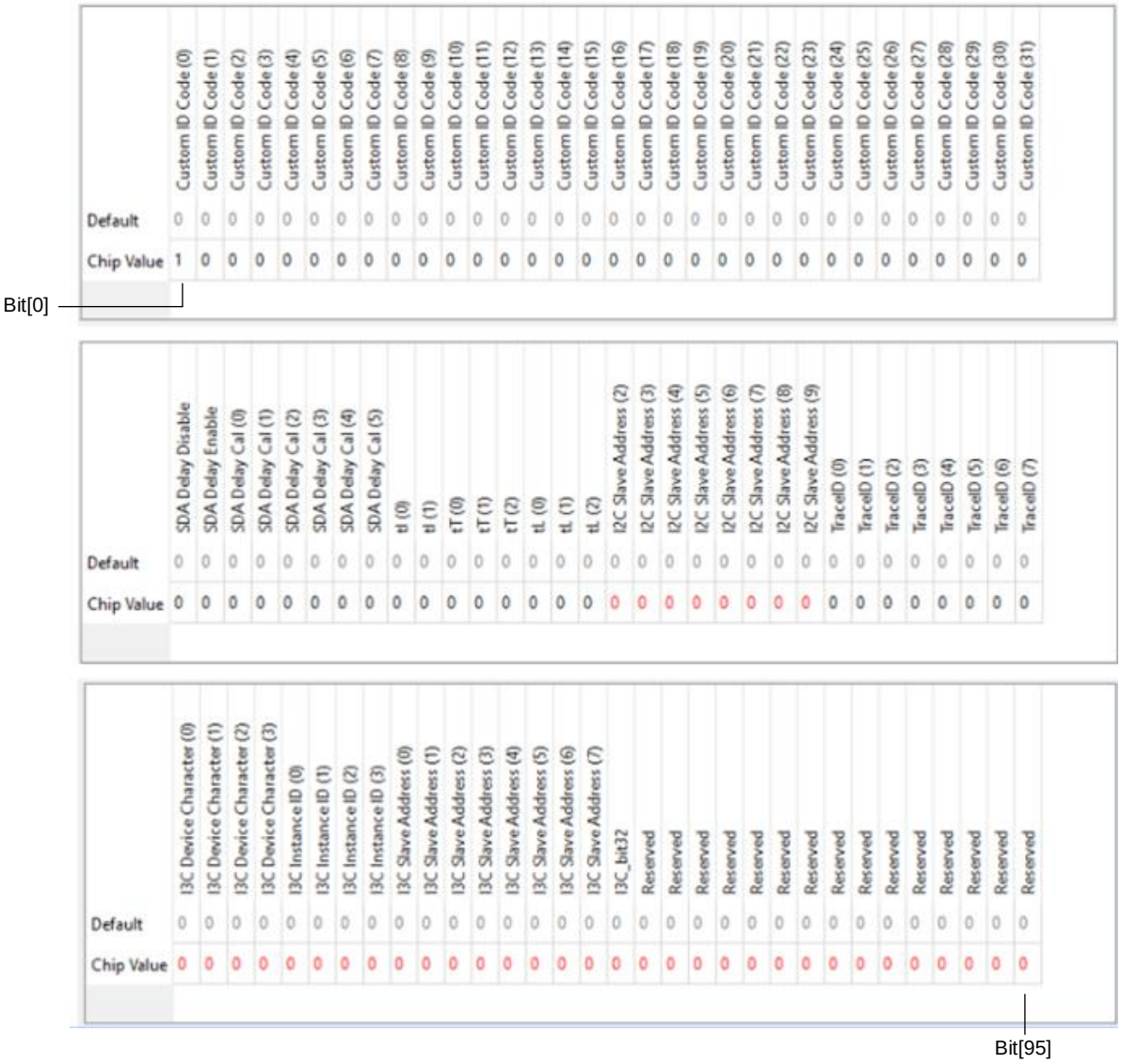


Figure C.2. 96-bit FEATURE Bits

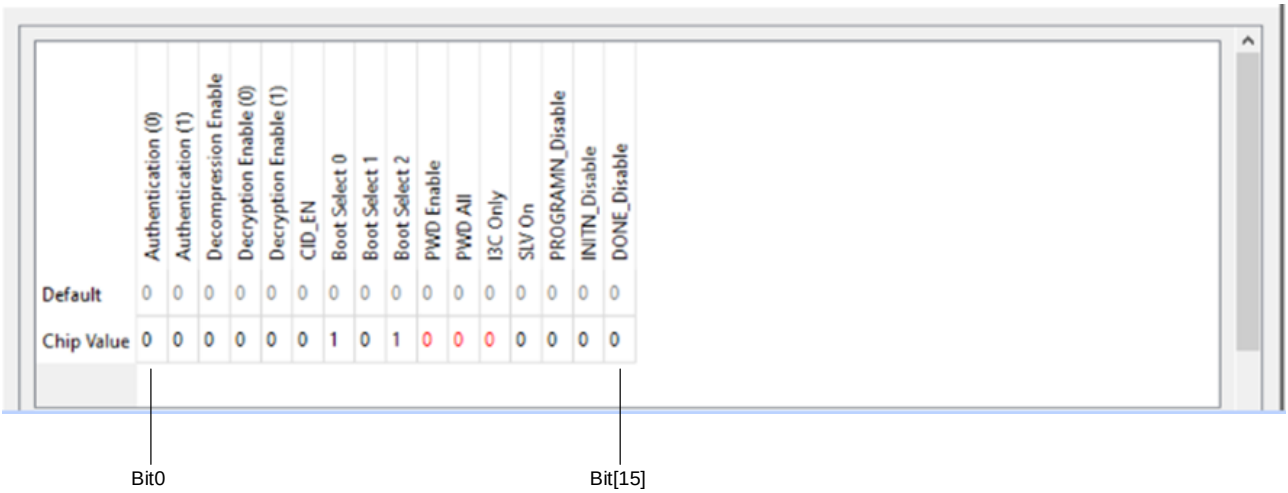


Figure C.3. 16-bit FEABITS

Modifying the Feature Row Programming File

You can use the Feature Row Editor available in the *Programming File Utility* to enable or disable silicon features in the Lattice FPGA devices to modify the Feature Row fuse settings in the data file (*.fea*).

The Programming File Utility tool can be accessed by using either of the following method:

- The *fileutility.exe* is located inside the *programmer* folder from the installation path of Lattice Radiant or a Lattice Radiant standalone programmer.
- Accessing through Radiant Programmer's *Tools* menu

Before modifying the feature row, you must create a project first in Lattice Radiant and run it through *Export Files*. The *.fea* file is generated inside the *Implementation* folder.

Next, edit the feature row bits by performing the steps below:

1. From the Programming File Utility, select **Tools > Feature Row Editor**. The Feature Row Editor dialog box opens.

   Using the **Browse** button, select the data (*.fea*) file to modify. Click **Read** to display the current data file settings. The software displays the default values in the top row. The second row shows the feature row bits in black that can be modified. The feature row bits that cannot be modified are shown in red.
2. Select the desired bit cell value to change and then toggle the value from 1 to 0, or from 0 to 1.
3. Click **Save** to overwrite the existing data file (*.fea*).
4. To create a new data file (*.fea*) with a different name, select **Save As**.

Modifying the Feature Row of a Programmed Device

Modifying the Feature Row using the .fea Programming File

To modify the feature row using the .fea file, perform the following steps:

1. Run the Radiant Programmer.
2. Using the Run menu, select **Scan Device**.
3. Once the scanning of the device is done, select **Edit > Device Properties**.
4. Change the Device Operation using the following settings:
   a. **Target Memory – Non Volatile Configuration Memory**
   b. **Port Interface – JTAG** (or any of the available interfaces: **Slave SPI**, **I^2C**, or **I3C Bridge**)
   c. **Access Mode – Feature Rows Programming**
   d. **Operation – Program Feature Row**

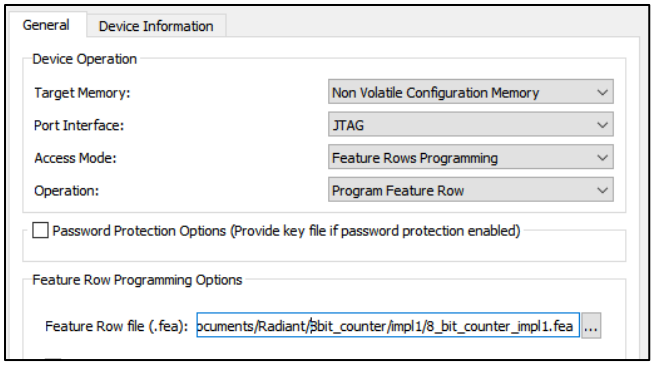


Figure C.4. Feature Row Programming Setup

5. Under Row Programming Options, select the *.fea* file from the *Implementation* folder.
6. Click **OK**, and then select **Run > Program Device**.

   Note: The feature row bits are OTP (One-Time Programmable) which can only be modified once.

Modifying the Feature Row without using the .fea Programming File

Updating the feature row can be also done without the need of the Lattice Radiant project or generating the *.fea* file.

To update the feature row, perform the following steps:

1. In the Radiant Programmer, user must select the device feature row to modify.
2. Choose **Edit > Device Properties**.
3. Change the Device Properties using the following settings:
   a. **Target Memory – Non Volatile Configuration Memory**
   b. **Port Interface – JTAG** (or any of the available interfaces: **Slave SPI**, **I²C**, or **I3C Bridge**)
   c. **Access Mode – Feature Rows Programming**
   d. **Operation – Update Feature Row**
4. Click **OK** to close the Device Properties dialog box.
5. Select **Design > Program**. The Feature Row dialog box displays.
6. Edit the fields in the dialog box as desired, and then click **Program**. The dialog box prompts the user to overwrite the selected feature row bit or bits. If the user selects **Yes**, the device's feature row is programmed.

   Note: Update Feature Row can only accept per-bit modification from 0 to 1. The modification is only applied to the values displayed in black.

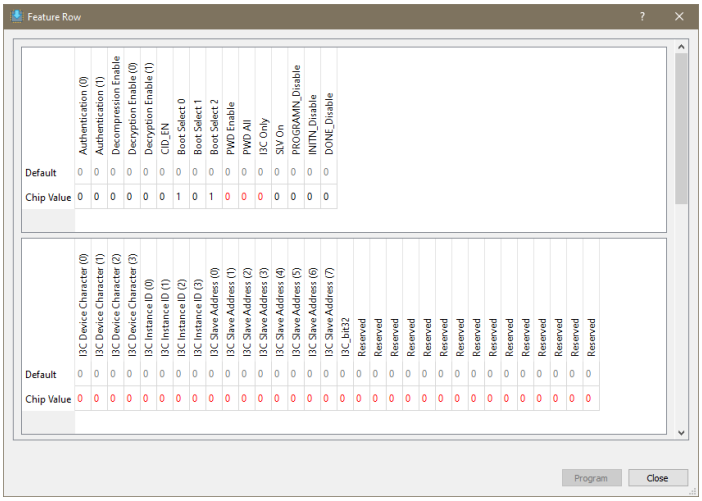


Figure C.5. Update Feature Row User Interface

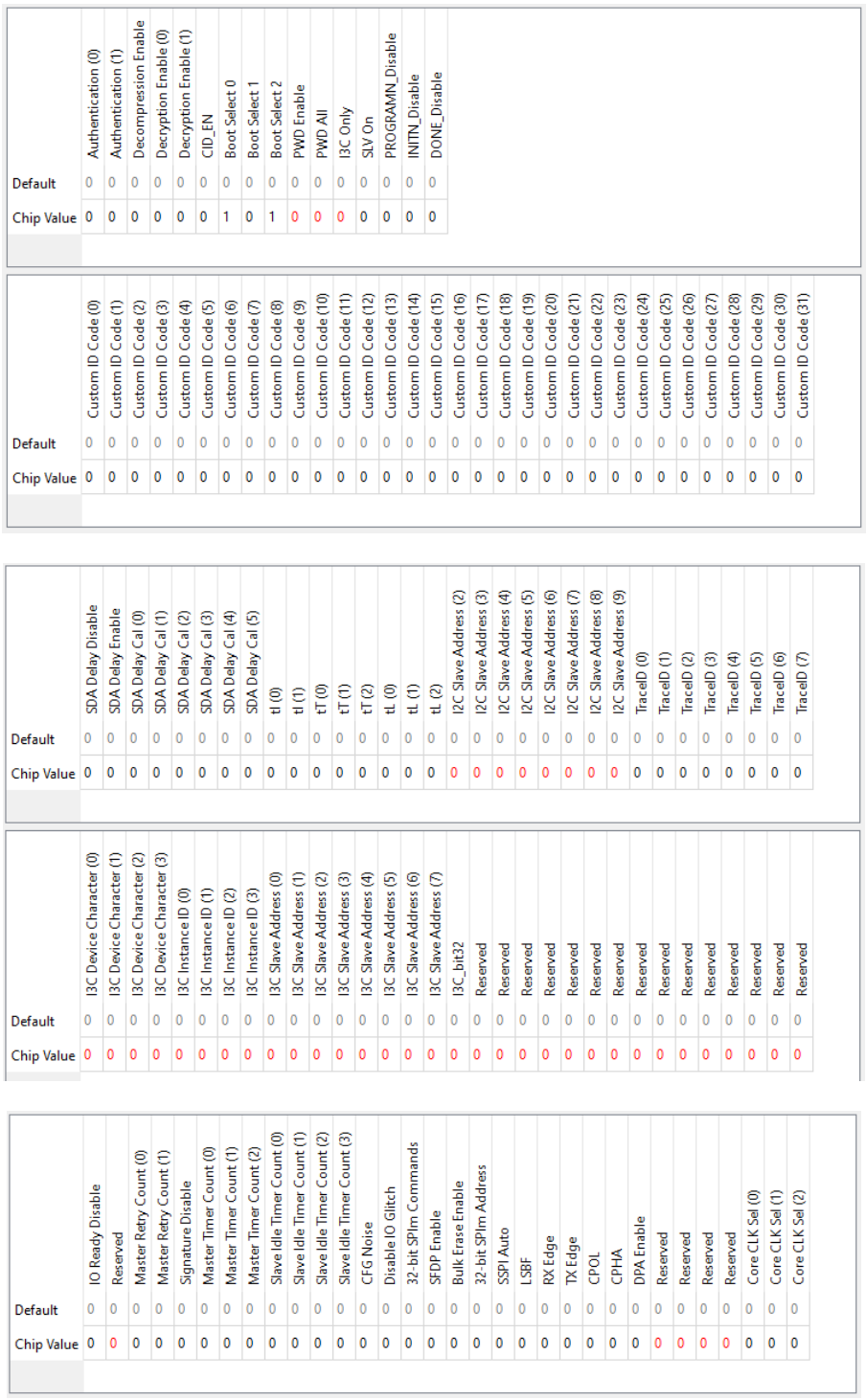


Figure C.6. Feature Row Bit Examples

Comparison of Feature Row Programming File (.fea)

Figure C.7 shows the 1:1 bit alignment of the .fea file (generated by the Lattice Radiant software) with regards to the feature row readout through the Radiant Programmer. For the individual bit definition, refer to the following:

- 32-bit NV CR1 bits – refer to Control Register 1 (CR1) section.
- 96-bit FEATURE bits – refer to Table 6.23 of the sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099).
- 16-bit FEABITS – refer to Table 6.24 of the sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099).

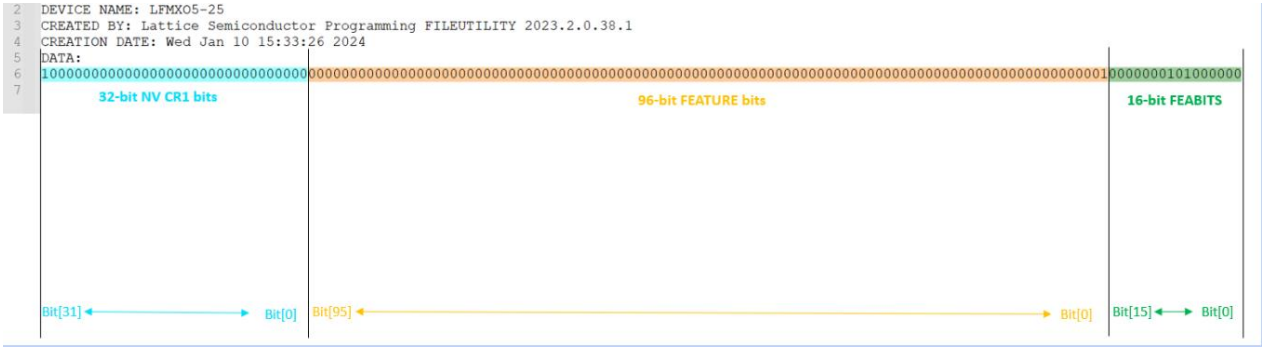


Figure C.7. .fea File Sample

References

For more information, refer to the following resources:

- MachXO5-NX Family Datasheet (FPGA-DS-02102)
- Minimizing System Interruption During Configuration Using TransFR Technology (FPGA-TN-02198)
- Flash Access IP Core (FPGA-IPUG-02171)
- Multi-Boot Usage Guide for Nexus Platform (FPGA-TN-02145)
- MachXO5-NX web page
- Development Kits & Boards for MachXO5-NX
- IP and Reference Designs for MachXO5-NX
- Lattice Radiant FPGA design software
- Lattice Propel FPGA design software
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, refer to the Lattice Answer Database at www.latticesemi.com/en/Support/AnswerDatabase.

Revision History

Revision 2.1, July 2024

Section	Change Summary
Acronyms in This Document	Added item.
Introduction	Added information on references for MachXO5-NX Root-of-Trust devices.

Configuration Details	• Added information for LFMXO5-15D and LFMXO5-55TD devices in Table 4.1. Maximum Configuration Bits of the Bitstream Sizes section.
	• In the INITN section of the sysCONFIG Pins section:
	  • Updated Figure 4.2. Configuration from PROGRAMN Timing.
	  • Updated Figure 4.3. Configuration Error Notification.
	  • Added information on restarting configuration after a configuration error as signaled by INITN going low during configuration.

Configuration Modes	• Updated Figure 6.3. Slave SPI Read Waveforms and Figure 6.4. Slave SPI Write Waveforms in the Specifications and Timing Diagrams – Slave SPI Port Waveforms section of the Slave SPI Mode section.
	• In the Class A Command Waveforms section of the Command Waveforms section:
	  • Updated description for bit shifted in first.
	  • Updated Figure 6.6. Class A Command Waveforms.
	• In the Class B Command Waveforms section of the Command Waveforms section:
	  • Updated description for bit shifted in first.
	  • Updated Figure 6.7. Class B Command Waveforms.
	• Added note for LSC_PROG_PASSWORD, LSC_PROG_CIPHER_KEY, LSC_PROG_FEABITS, LSC_PROG_OTP, and LSC_PROG_ECDSA_PUBKEY to indicate that operation programs internal EFUSE memory, which is one-time programmable, in Table 6.13. JTAG Instruction Table of the JTAG Instruction Support section.

Appendix B	• In the MachXO5-NX Device Specific section:
	  • Added information for LFMXO5-15D and LFMXO5-55TD devices in Table B.5. MachXO5-NX Device Specifics.
	  • In Table B.6. MachXO5-NX Device ID:
	  • Added LFMXO5-15D and LFMXO5-55TD devices.
	  • Added note to table regarding special API requirement to read the device ID of MachXO5-NX Root-of-Trust devices.

Revision 2.0, May 2024

Section	Change Summary

Configuration Modes	• Minor editorial changes.
	• Figure 6.5. Slave SPI Configuration Flow in Slave SPI Configuration Flow Diagrams section:
	  • Updated flow.
	  • Added note 1 regarding PROGRAMN, INITN, and DONE pins.
	  • Added note 2 with cross reference to Table 6.5. Execution Time for ISC_ERASE Command.
	  • Added note 5 regarding external DONE pin and wait time before checking the status register.
	  • Added note 6 to indicate that AUTH DONE=1 check only applicable if encrypted bitstream is used.
	• Added Table 6.5. Execution Time for ISC_ERASE Command in Slave SPI Configuration Flow Diagrams section.
	• Updated CCLK waveform in Figure 6.6. Class A Command Waveforms in Class A Command Waveforms section.
	• Updated CCLK waveform in Figure 6.7. Class B Command Waveforms in Class B Command Waveforms section.

Section	Change Summary

	• Updated CCLK waveform in Figure 6.8. Class C Command Waveforms in Class C Command Waveforms section.
	• Updated CCLK waveform in Figure 6.9. Class D Command Waveforms in Class D Commands Waveforms section.
	• Added note on reference for slave I2C/I3C configuration flow in Slave I2C/I3C Mode section.
	• Updated reference to TransFR document in TransFR Operation section.
	• In SPI/I2C/I3C Command Support section:
	  • Removed the Slave Configuration Interface Command Table.
	  • Added reference to the sysCONFIG User Guide for Nexus Platform (FPGA-TN-02099) for ISC and programming commands for non-JTAG configuration interfaces.
	• Updated JTAG instructions to "LSC_PROG_ECDSA_PUBKEY" and "LSC_READ_ECDSA_PUBKEY" in Table 6.13. JTAG Instruction Table in JTAG Instruction Support section.

Flash Programming	Added discussion on hard JTAG port limitation during flash access operation in Flash Access through LMMI Bus section.
Software Selectable Options	• Updated Figure 8.1. sysCONFIG Preferences in Global Tab, Lattice Radiant Device Constraint Editor. • Updated TRANSFR default setting in Table 8.1. sysCONFIG Options. • Updated reference to TransFR document in TRANSFR section.
References	Updated reference to TransFR document.

Revision 1.9, February 2024

Section	Change Summary

Configuration Modes	• Added SSPI persistence information in Slave SPI Mode.
	• Updated the following in Table 6.12. Slave Configuration Interface Command Table:
	  • Added footnote and reference to footnote 7 in LSC_PROG_FEATURE and LSC_PROG_FEABITS.
	  • Removed reference to footnote 7 in LSC_AUTH_CTRL, LSC_PROG_ECDSA_PUBKEY, and LSC_READ_ECDSA_PUBKEY.
	• Added footnote and reference to footnote for LSC_PROG_FEATURE and LSC_PROG_FEABITS in Table 6.13. JTAG Instruction Table.

Appendix C. Modifying the Nexus Feature Row	Added Breakdown of Feature Row Bits and Comparison of Feature Row Programming File (.fea) sections.

Revision 1.8, January 2024

Section	Change Summary
Configuration Modes	• Updated all references of the term “delay time” to execution time” in this section.
Configuration Process and Flow	• Updated a sentence in 5.1 Power-up Sequence section: *During the power-up sequence phase, once the POR circuit is active, the POR circuit makes sure that the external I/O pins are in a high impedance state.*
Flash Programming	• Added the 7.1 Flash Memory Space Partition section. • Added the 7.7 Ping-Pong Boot section. • Added the 7.8 Enabling Fast Configuration section.
Software Selectable Options	• Updated the 8.17 BOOTMODE section. • Updated the 8.19 CONFIGURATION section. • Updated the 8.20 CUR_DESIGN_BOOT_LOCATION section. • Updated the 8.21 PRIMARY_BOOT section. • Updated the 8.22 SECONDARY_BOOT section.

Revision 1.7, December 2023

Section	Change Summary
Configuration Details	• Added SD[15:4] sysCONFIG Pin in Table 4.4. Default State of the sysCONFIG Pins. • Added a note for SD[15:4], *Reserved when the device is in configuration mode,* in Table 4.4.

Revision 1.6, October 2023

Section	Change Summary
Configuration Modes	Added the statement, *After the internal DONE bit is asserted the GPIO will wake-up, the external DONE pin will go high after 10 µs. This means that the GPIO pin settings will be applied before the DONE pin goes high,* to section 4.4.3 DONE.

Revision 1.5, September 2023

Section	Change Summary
Configuration Modes	Removed the JUMP command in Table 6.12.

Revision 1.4, August 2023

Section	Change Summary
Configuration Details	Changed the statement *from an external memory* to *from internal flash memory*.
References	Added this section.

Revision 1.3, April 2023

Section	Change Summary
Configuration Details	Added a note to the INITN section.

Revision 1.2, April 2023

Section	Change Summary
Configuration Details	Updated Table 4.1. Maximum Configuration Bits to add LFMXO5-55T and LFMXO5-100T devices.
Appendix B. MachXO5-NX Bitstream File Format	Updated Table B.5. MachXO5-NX Device Specifics and Table B.6. MachXO5-NX Device ID to add LFMXO5-55T and LFMXO5-100T devices.
Technical Support Assistance	Added reference link to the Lattice Answer Database.

Revision 1.1, October 2022

Section	Change Summary
All	Minor adjustments in formatting across the document.

Configuration Details	• Updated sysCONFIG Pins to change the following:
	  • Changed VCCIO0 and VCCIO1 to VCCIO1 and VCCIO2.
	  • Changed Bank 0 and Bank 1 to Bank 1 and Bank 2.
	• Changed VCCIO0 to VCCIO1 in PROGRAMN.

Configuration Process and Flow	• Updated Figure 5.1. Configuration Flow to change VCCIO0 to VCCIO2. • Updated Power-up Sequence to change VCCIO0 to VCCIO2.
Software Selectable Options	• Updated Table 8.1. sysCONFIG Options to change CONFIGIO_VOLTAGE_BANK0 to CONFIGIO_VOLTAGE_BANK2. • Updated CONFIGIO_VOLTAGE_BANK1/CONFIGIO_VOLTAGE_BANK2 to change CONFIGIO_VOLTAGE_BANK0/CONFIGIO_VOLTAGE_BANK1 to CONFIGIO_VOLTAGE_BANK1/CONFIGIO_VOLTAGE_BANK2, and Bank 0 and Bank 1 to Bank 1 and Bank 2.
Appendix C. Modifying the Nexus Feature Row	Added this section.

Revision 1.0, May 2022

Section	Change Summary
All	Initial production release for MachXO5-NX device family.

www.latticesemi.com