Intel Stratix 10 Configuration User Guide

Updated for Intel® Quartus® Prime Design Suite: **18.1**

Contents

1. Intel® Stratix® 10 Configuration Overview

1.1. Intel® Stratix® 10 Configuration Overview

All Intel® Stratix® 10 devices include a secure device manager (SDM) to manage FPGA configuration and security. The SDM provides a failsafe, strongly authenticated, programmable security mode for device configuration. Previous FPGA families include a fixed state machine to manage device configuration.

The Intel Quartus® Prime software also provides flexible and robust security features to protect sensitive data, intellectual property, and the device itself under both remote and physical attacks. Configuration bitstream authentication ensures that the firmware and configuration bitstream are from a trusted source. Encryption prevents theft of intellectual property. The Intel Quartus Prime software also compresses FPGA bitstreams, reducing memory utilization.

Intel describes configuration schemes from the point-of-view of the FPGA. Intel Stratix 10 devices support active and passive configuration schemes. In active configuration schemes the FPGA acts as the master and the external memory acts as a slave device. In passive configuration schemes an external host acts as the masters and controls configuration. The FPGA acts as the slave device. All Intel Stratix 10 configuration schemes support design security, remote system upgrade, and partial reconfiguration. To implement remote system update in passive configuration schemes, an external controller must store and drive the configuration bitstream.

Intel Stratix 10 devices support the following configuration schemes:

- Avalon® Streaming (Avalon-ST)
- JTAG
- Configuration via Protocol (CvP)
- Active Serial (AS) normal and fast modes
- Secure Digital and Multi Media Card (SD MMC)

Table 1. Intel Stratix 10 Configuration Data Width, Clock Rates, and Data Rates

Mbps is an abbreviation for Megabits per second.

Configuration Scheme		Data Width (bits)	MSEL[2:0]
Passive	Avalon-ST	32	000
		16	101
		8	110
	JTAG	1	111
	Configuration via Protocol (CvP)	x1, x2, x4, x8, x16 lanes	001

continued...

Configuration Scheme		Data Width (bits)	MSEL[2:0]
Active	SD MMC	4/8	100
	AS - fast mode	4	001
	AS - normal mode	4	011

Avalon-ST

The Avalon-ST configuration scheme is a new passive configuration scheme for Intel Stratix 10 devices. It replaces the fast passive parallel (FPP) mode available in earlier device families. Avalon-ST is the fastest configuration scheme for Intel Stratix 10 devices. Avalon-ST configuration supports x8, x16, and x32 modes. The x16 and x32 bit modes use general-purpose I/Os (GPIOs) for configuration. You can repurpose these GPIOs after configuration completes. The x8 bit mode uses dedicated SDM I/O pins.

Avalon-ST differs from FPP configuration in supporting backpressure using the AVST_READY and AVST_VALID pins. Because the time to decompress the incoming bitstream varies, backpressure support is necessary to transfer data to the Intel Stratix 10 device. For more information about the Avalon-ST refer to the *Avalon Interface Specifications*.

JTAG

You can configure the Intel Stratix 10 device using the dedicated JTAG pins. The JTAG port provides seamless access to many useful tools and functions. In addition to programming memories JTAG port is useful for debugging using Signal Tap or the System Console tools.

The JTAG port has the highest priority and overrides the MSEL pin settings. Consequently, you can configure the Intel Stratix 10 device over JTAG even if the MSEL pin specifies a different configuration scheme unless you disabled JTAG for security reasons.

CvP

CvP uses an external PCIe* host device as a Root Port to configure the Intel Stratix 10 device over the PCIe link. You can specify up to a x16 PCIe link. Intel Stratix 10 devices support two CvP modes, CvP init and CvP update.

CvP initialization process includes the following two steps:

1. CvP configures the FPGA periphery image which includes I/O information and hard IP blocks, including the PCIe IP. Because the PCIe IP is in the periphery image, PCIe link training establishes PCIe link of the CvP PCIe IP before the core fabric configures.
2. Then, the host device uses the CvP PCIe link to configure your design in the core fabric.

CvP update mode updates the FPGA core image the using the PCIe link already established from a previous full chip configuration or CvP init configuration. After the Intel Stratix 10 enters user mode, you can use the CvP update mode to reconfigure the FPGA fabric. This mode has the following advantages:

- Allows reprogramming of the core to run different algorithms
- Provides a mechanism for standard updates as a part of a release process.
- Customizes core processing for different components that are part of a complex system

  For both CvP Init and CvP Update modes, the maximum data rate depends on the PCIe generation and number of lanes.

For more information refer to the *Intel Stratix 10 Configuration via Protocol (CvP) Implementation User Guide* .

AS Normal Mode

Active Serial or AS x4 or Quad SPI (QSPI) is an active configuration scheme that supports flash memories capable of three- and four-byte addressing. The configuration firmware requires three-byte addressing. After the configuration firmware loads, the AS x4 flash uses four-byte addressing for the rest of the configuration process. This mode supports Intel's serial flash configuration memory solution the following third-party flash devices:

- Micron MT25Q 512 megabytes (MB)
- Macronix MX66U 512 MB, 1 and 2 gigabytes (GB)
- Macronix MX25U 128 MB, 256 MB, and 512 MB
- Micron MT25QU 128 MB, 256 MB, 512 MB, 1 GB, and 2 GB

Refer to the *Supported CFI Flash Memory Devices* appendix for a complete list of supported flash devices.

AS Fast Mode

The only difference between AS normal mode and fast mode is speed. Use AS fast mode when configuration timing is a concern. Use this mode to meet the 100 ms of power up requirement for PCIe or for other systems with strict timing requirements.

In AS fast mode, the SDM first powers the external AS x4 flash. The power supply must be able to provide an equally fast ramp up for the Intel Stratix 10 device and the external AS x4 flash devices. Failing to meet this requirement causes the SDM to assume missing memory. Consequently, configuration fails.

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* and *AN692: Power Sequencing Considerations for Intel Cyclone® 10 GX, Intel Arria® 10, and Intel Stratix 10 Devices* for additional details

SD MMC

SD MCC is an active configuration scheme. The Intel Stratix 10 SDM can initiate configuration from the SD MCC cards. The SD MMC mode is almost identical to AS x4. The difference is that SD MMC are removable and follow a standard protocol. The advantages of this mode are cost, capacity, availability, portability, and compatibility. Because Intel Stratix 10 devices operate at 1.8 volt and SD MMC I/Os operate between 2.7 - 3.6 volts, an intermediate voltage level translator is necessary.

Note: The SD MMC configuration scheme is not supported in the current release.

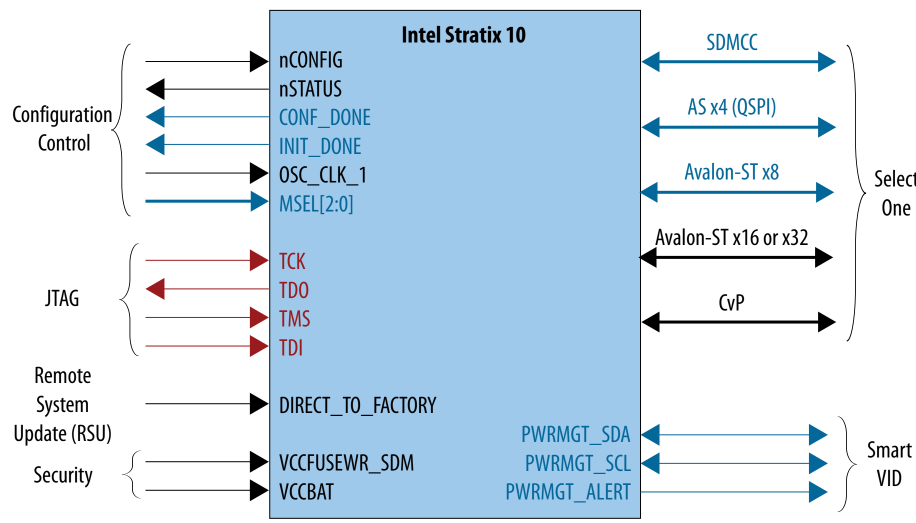
Related Information

- Supported CFI Flash Memory Devices on page 111
- Intel Stratix 10 GX and SX Device Family Pin Connection Guidelines
- AN 692: Power Sequencing Considerations for Intel Cyclone 10 GX, Intel Arria 10, and Intel Stratix 10 Devices
- Device Configuration - Support Center
- Avalon Interface Specifications
- Intel Stratix 10 Configuration via Protocol (CvP) Implementation User Guide
- Intel Stratix 10 Device Datasheet (Core and HPS)

1.1.1. Configuration and Related Signals

The following figure shows the configuration interfaces and configuration-related device functions. Pins shown in dark blue use dedicated SDM I/Os. Pins shown in black use general purpose I/Os (GPIOs). Pins shown in red are dedicated JTAG I/Os. You specify SDM I/O pin functions using the **Device ➤ Configuration ➤ Device and Pin Options** dialog box in the Intel Quartus Prime software.

Figure 1. Intel Stratix 10 Configuration Interfaces



This user guide discusses most of the interfaces shown in the figure. Refer to the separate *Intel Stratix 10 Configuration via Protocol (CvP) Implementation User Guide* and *Intel Stratix 10 Power Management User Guide* for more information about those features.

Related Information

- SDM Pin Mapping on page 17
- Intel Stratix 10 Configuration via Protocol (CvP) Implementation User Guide
- Intel Stratix 10 Power Management User Guide

1.1.2. Intel Download Cables Supporting Configuration in Intel Stratix 10 Devices

Intel provides the following cables to download your design to the Intel Stratix 10 device on the PCB. Download cables support prototyping activity by providing detailed debug messages via Intel Quartus Prime Programmer. You must use Intel download cables for advanced debugging using the Signal Tap logic analyzer the System Console.

Table 2. Intel Stratix 10-Supported Download Cable Capabilities

Download Cable	Protocol Support Intel Stratix 10 Device	Cable Connection to PCB
Intel FPGA Download Cable II (formerly the USB-Blaster II)	JTAG, AS	10-pin female plug 3M Part number: 2510-6002UB
Intel FPGA Ethernet Cable (formerly the EthernetBlaster II)	JTAG, AS	10-pin female plug

For more information about download cables refer to Intel FPGAs and Programmable Devices / Download Cables. This web page includes links to the user guides for all the cables listed in Table 2 on page 8.

1.2. Intel Stratix 10 Configuration Architecture

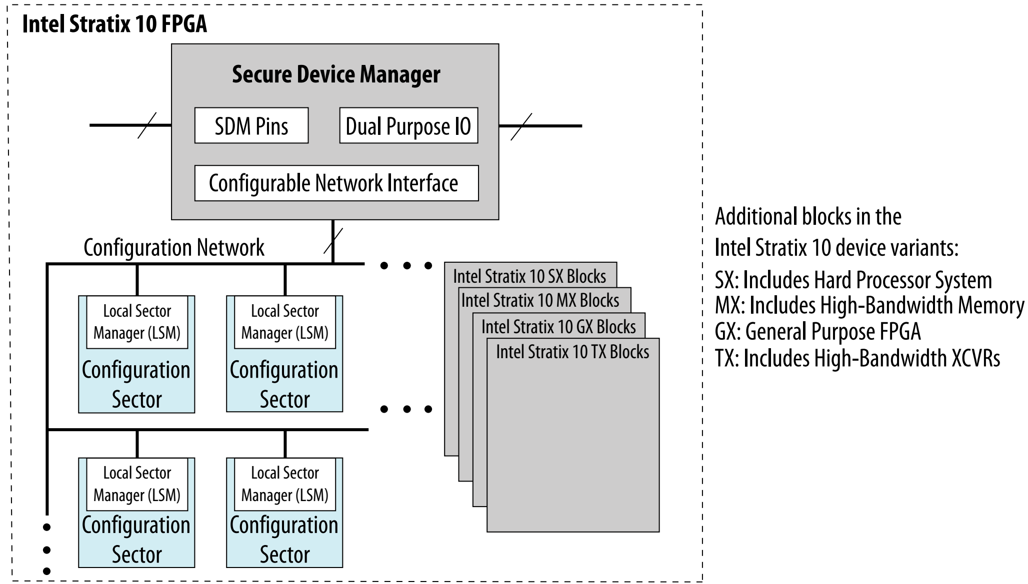
The Secure Device Manager (SDM) is a triple-redundant processor-based module that manages configuration and the security features of Intel Stratix 10 devices. The SDM is available on all Intel Stratix 10 FPGAs and SoC devices.

The block diagram below provides an overview of the Intel Stratix 10 configuration architecture which includes the following blocks:

- Secure device manager (SDM): More information about SDM is contained in later sections.
- Configuration network: The SDM uses this dedicated, parallel configuration network to distribute the configuration bitstream to Local Sector Managers (LSMs). You cannot access this network.
- LSMs: The LSM is a microprocessor. Each configuration sector includes an LSM. The LSM parses configuration bitstream and configures the logic elements for its sector. After configuration, the microprocessors perform the following functions:
  - Monitors for single event upsets at the sector level
  - Processes responses to SEUs
  - Performs hashing or integrity checks in real time
- Specific blocks for Intel Stratix 10 variants:
  - SX devices include the hard processor system (HPS) in addition to FPGA logic.
  - MX devices include a High Bandwidth Memory (HBM) in addition to FPGA logic.
  - GX devices include FPGA logic and L- and H-Tile transceivers.

    TX devices include FPGA logic and E- and H-Tile transceivers.

Figure 2. Intel Stratix 10 Configuration Architecture Block Diagram

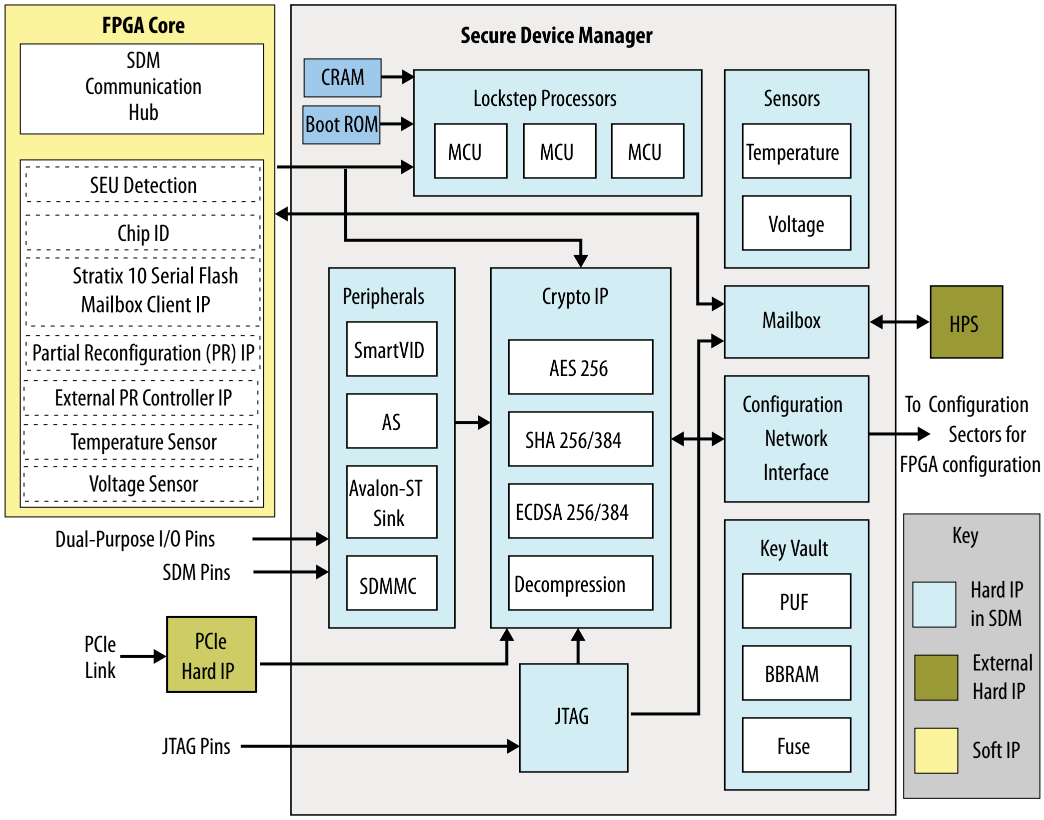


1.2.1. Secure Device Manager

The SDM comprises peripherals, cryptographic IP and sensors, boot ROM, triple-redundant lockstep processors, and other blocks shown the block diagram below. The SDM performs and manages the following security functions:

- Configuration bitstream authentication: After power-on during startup, the SDM triple-redundant lockstep processors run code from the boot ROM. The boot ROM code authenticates the Intel-generated configuration bitstream, ensuring that configuration bitstream is from a trusted source.
- Encryption: Encryption protects the configuration bitstream or confidential data from unauthorized third-party access.
- Side channel attack protection: Side channel attack protection prevents AES Key and confidential data under non-intrusive attacks.
- Integrity checking: Integrity checking verifies that an accidental event has not corrupted the configuration bitstream. This function is active, even if you do not enable authentication.

Figure 3. SDM Block Diagram



Here is an overview of the additional functions the SDM controls:

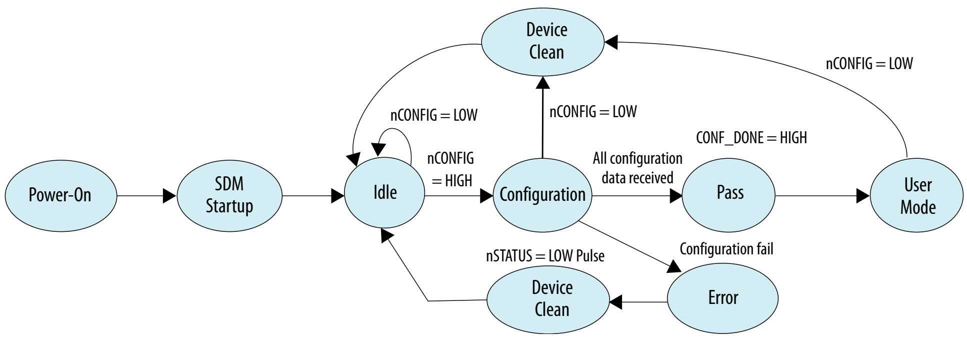
- The Power Management block consists of a voltage and temperature sensor which enables the SmartVID feature via an external PMBus voltage regulator when you select -V devices.
- The AES/SHA and other Crypto Accelerator blocks implement secure configuration and boot.
- The Key Vault provides volatile and non-volatile cryptographic key storage. To mitigate potential side-channel attacks, crypto functions that use keys require the keys a special hardware storage mechanism. For more information refer to the *Intel Stratix 10 Device Security User Guide*.
- The AS and SD MMC configuration flash controllers enable active configuration schemes via dedicated SDM pins.
- The x8 Avalon-ST configuration scheme SDM I/O pins. The x16 and x32 Avalon-ST configuration schemes use dedicated SDM I/O Pins and dual-purpose I/O pins. Refer to the *SDM Pin Mapping* for more information.
- To reduce configuration file size and support smaller memory sizes, and enable faster configuration, the Intel Quartus Prime software compresses the configuration data. Intel Stratix 10 devices all compress the configuration bitstream. You cannot disable this feature. The decompression block in the SDM decompresses both encrypted and non-encrypted configuration files.
- A specific PCIe block included in the Intel Stratix 10 device supports CvP.

Related Information

SDM Pin Mapping on page 17

2. Intel Stratix 10 Configuration Details

2.1. Configuration Flow Diagram



The following section provides information about each configuration state:

Power Up

- Power up as per specifications in the *Intel Stratix 10 Power Management User Guide*.
- The V_{CCERAM} is the first power, V_{CCR} is second, and V_{CCIO_SDM} is last.
- Device wide Power on Reset (POR) asserts after power supplies reach operating voltage. POR deasserts after reach trip-out voltage. The external power supply ramp must not be slower than the minimum ramping rate until the supplies reach the operating voltage.
- Most SDM_IO pins remain pulled weak high internally. The SDM_IO0, SDM_IO8, SDM_IO16 pins remain low internally.

SDM Startup

- SDM runs firmware stored in the SDM internal boot ROM.
- Boot ROM authenticates Intel-generated configuration bitstream.
- The SDM samples the MSEL pins during power-on.
- If MSEL is set to JTAG, the SDM remains in the Startup state.

Idle

- The SDM remains in IDLE state until the external host initiates configuration by driving the nCONFIG pin from low to high, alternatively, the SDM enters the idle state after it exits the error state.
- The SDM reads the nCONFIG pin status.
- If the nCONFIG pin is high and a configuration error occurs, the SDM drives nSTATUS pin low for 1 ms ±50%. After which nSTATUS remains high until an external source drives the nCONFIG pin low. After nCONFIG is driven low, the SDM drives nSTATUS low. The SDM is now in the idle state.

Configuration Start

- The SDM receives configuration data from the source configuration bitstream and performs authentication, decryption and decompression.
- The SDM configures the FPGA fabric.
- Configuration begins when the nSTATUS pin is high.
- The nCONFIG pin remains high during configuration and in user mode.

Configuration Pass

- The SDM enters the Pass state after it completes device configuration.
- The Intel Stratix 10 device drives the CONF_DONE or INIT_DONE pin high after successful configuration completes.
- The device is not in user mode.

Configuration Error

- A low put on the nSTATUS pin LOW for 1ms ± 50% indicates a configuration error.
- After low pulse indicating an error, the SDM drives the nSTATUS pin high if the nCONFIG pin remains high. These unique pin conditions indicate a configuration error.
- After an error the SDM enters Idle state if nCONFIG is low until the external host drives the nCONFIG pin high.

User Mode

- The SDM drives the INIT_DONE pin high after initializing internal registers and releases GPIO pins from the high impedance state. The device enters user mode. The entire device does not enter user mode at the same instant.
- The nCONFIG pin should remain high in user mode. To initiate reconfiguration, the external host drives nCONFIG low. Then, a rising edge on nCONFIG initiates reconfiguration.

Device Clean

- Device cleaning zeros out all configuration data.
- The design stops functioning.
- The Intel Stratix 10 device drives CONF_DONE and INIT_DONE low.

  The nSTATUS pin goes low when device cleaning completes.

JTAG Configuration

Note: You can perform JTAG configuration anytime from any state except the power-on and SDM startup state. The Intel Stratix 10 device cancels the previous configuration and accepts the reconfiguration data from the JTAG interface. The nCONFIG signal must be held in a stable or low state during JTAG configuration. A falling edge on the nCONFIG signal cancels the JTAG configuration.

Note: The SDM only samples the MSEL pins at power-on and initiates bitstream configuration using the configuration scheme specified at power-on.

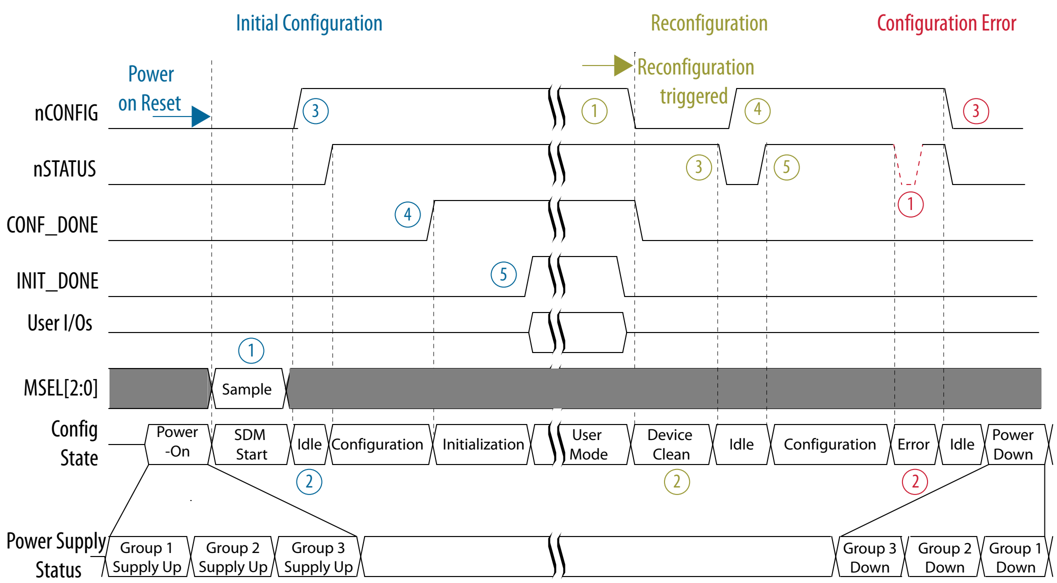
Related Information

Booting and Configuration in the Intel Stratix 10 Hard Processor System Technical Reference Manual

2.2. Intel Stratix 10 Configuration Timing Diagram

The SDM drives Intel Stratix 10 device configuration.

Figure 4. Configuration, Reconfiguration and Error Timing Diagram



Initial Configuration Timing

The first section of the figure shows the expected timing for initial configuration after a normal power-on reset. The initial state of the nCONFIG and nSTATUS signals is low.

The numbers in the *Initial Configuration* part of the timing diagram mark the following events:

1. The SDM boots up and samples the MSEL signals to determine the specified FPGA configuration scheme. The SDM does not sample the MSEL pins again until the next power cycle.
2. With the nCONFIG signal low, the SDM enters Idle mode after booting.
3. When the external host drives nCONFIG signal high, the SDM initiates configuration. The SDM drives the nSTATUS signal high, signaling the beginning of FPGA configuration. The SDM receives the configuration bitstream on the interface that MSEL value sampled in *Step 1*.
4. The SDM drives the CONF_DONE signal high, indicating successful configuration.
5. When the Intel Stratix 10 device asserts INIT_DONE the FPGA enters user mode. GPIO pins exit the high impedance state. The entire device does not enter user mode at the same instant.

Reconfiguration Timing

The second event the timing diagram illustrates the Intel Stratix 10 device reconfiguration. Note that the reconfiguration assumes that you have not changed the MSEL setting. If you do change the MSEL setting after power-on, you must power-cycle the Intel Stratix 10 device. Power cycling forces the SDM to sample the MSEL pins before reconfiguring the device.

The numbers in the *Reconfiguration* part of the timing diagram mark the following events:

1. The external host drives nCONFIG signal low.
2. The SDM initiates device cleaning.
3. The SDM drives the nSTATUS signal low when device cleaning is complete.
4. The external host drives the nCONFIG signal high to initiate reconfiguration.
5. The SDM drives the nSTATUS signal high signaling the device is ready for reconfiguration.

Configuration Error

The numbers in the *Configuration Error* part of the timing diagram mark the following events:

1. The SDM drives nSTATUS signal low for 1 ms ±50% to indicate a configuration error. The Intel Stratix 10 devices does not assert CONF_DONE indicating that configuration did not complete successfully.
2. The SDM enters the error state.
3. The SDM enters the idle state. The external host deasserts nCONFIG. The device is ready for reconfiguration by driving a low to high transition on nCONFIG. You can also power cycling the device by following the device power down sequence.

Power Supply Status

The power-on reset (POR) holds the Intel Stratix 10 device in the reset state until the power supply outputs are within the recommended operating range. t_{RAMP} defines the maximum power supply ramp time. If POR does not meet the t_{RAMP} time, the Intel Stratix 10 device I/O pins and programming registers remain tri-stated.

For more information about POR refer to the *Intel Stratix 10 Power Management User Guide.* For more information about t_{RAMP} refer to the *Intel Stratix 10 datasheet*.

Related Information

- Intel Stratix 10 Power Management User Guide
- Intel Stratix 10 Device Datasheet (Core and HPS)
- Should clocks and resets in user logic be gated until the configuration process is completed in Intel Stratix 10?

2.3. Additional Clock Requirements for Transceivers, HPS, PCIe, High Bandwidth Memory (HBM2) and SmartVID.

The Intel Stratix 10 device has additional clock requirements for transceivers, the HPS, PCIe, SmartVID, and the High Bandwidth Memory (HBM2) IP.

Follow these guidelines to ensure successful device configuration and reconfiguration:

- For designs including the High Bandwidth Memory (HBM2) IP or any IP using transceivers, you must provide a free running and stable reference clock to the device before device configuration begins. All transceiver power supplies must be at the required voltage.

  Note: The transceivers must have their own power supplies. You can use the V_{CC} and V_{CCP} power supplies for initial transceiver testing. However, eventually device configuration fails because transceiver calibration cannot complete.

- For the HPS, the HPS clock and HPS DDR clock must be present and stable before configuration begins.
- For SmartVID devices, refer to the *Intel Stratix 10 Power Management and VID Interface Implementation Guide* chapter in the *Intel Stratix 10 Power Management User Guide*. This chapter provides instructions on assigning the **VID Operation mode**, the PMBus mode pins, PWRMGT_SCL, PWRMGT_SDA, and PWRMGT_ALERT, and the required software settings.
- Designs that use any transceivers on the Intel Stratix 10 device must provide an external, free-running, stable reference clock input to the OSC_CLK_1 pin before device configuration begins.

2.4. Intel Stratix 10 Configuration Pins

Intel Stratix 10 uses SDM pins for device configuration.

2.4.1. SDM Pin Mapping

You can use SDM pins for configuration and other functions; for example, power management. You specify SDM I/O pin functions using the **Device ➤ Configuration ➤ Device and Pin Options** dialog box in the Intel Quartus Prime software. Refer to the *Intel Stratix 10 Device Pinouts* and *Intel Stratix 10 Pin Connection Guidelines* for more details on other functions.

Table 3. SDM Pin Mapping

SDM Pins	MSEL Function	Configuration Source Function			Select Other Functions
		Avalon-ST x8	AS	SD/MMC	
SDM_IO0	—	—	—	—	INIT_DONE PWRMGT_SCL PWRMGT_ALERT DIRECT_TO_FACTORY SEU_ERROR
SDM_IO1	—	AVSTx8_DATA2	AS_DATA1	SDMMC_CFG_DATA1	—
SDM_IO2	—	AVSTx8_DATA0	AS_CLK	SDMMC_CFG_DATA0	—
SDM_IO3	—	AVSTx8_DATA3	AS_DATA2	SDMMC_CFG_DATA2	—
SDM_IO4	—	AVSTx8_DATA1	AS_DATA0	SDMMC_CFG_CMD	—
SDM_IO5	MSEL0	—	AS_nCSO0	SDMMC_CFG_CCLK	CONF_DONE, INIT_DONE
SDM_IO6	—	AVSTx8_DATA4	AS_DATA3	SDMMC_CFG_DATA3	—
SDM_IO7	MSEL1	—	AS_nCSO2	—	—
SDM_IO8	—	AVST_READY[1]	AS_nCSO3	SDMMC_CFG_DATA4	—
SDM_IO9	MSEL2	—	AS_nCSO1	—	—
SDM_IO10	—	AVSTx8_DATA7	—	SDMMC_CFG_DATA7	DIRECT_TO_FACTORY SEU_ERROR
SDM_IO11	—	AVSTx8_VALID	—	—	PWRMGT_SDA DIRECT_TO_FACTORY SEU_ERROR
SDM_IO12	—	—	—	—	PWRMGT_SDA PEWRMGT_ALERT DIRECT_TO_FACTORY

continued...

(1) AVST_READY is applicable in Avalon-ST x8, x16 and x32 configuration schemes.

SDM Pins	MSEL Function	Configuration Source Function			Select Other Functions
		Avalon-ST x8	AS	SD/MMC	
SDM_IO13	—	AVSTx8_DATA5	—	SDMMC_CFG_DATA5	DIRECT_TO_FACTORY SEU_ERROR
SDM_IO14	—	AVSTx8_CLK	—	—	PWRMGT_SDA DIRECT_TO_FACTORY
SDM_IO15	—	AVSTx8_DATA6	—	SDMMC_CFG_DATA6	DIRECT_TO_FACTORY SEU_ERROR
SDM_IO16	—	—	—	—	CONF_DONE INIT_DONE PWRMGT_SDA DIRECT_TO_FACTORY SEU_ERROR

Related Information

Intel Stratix 10 Device Pinouts

2.4.2. MSEL Settings

The MSEL[2:0] pins set the configuration scheme for Intel Stratix 10 devices. Use 4.7-kΩ resistors to pull the MSEL[2:0] pins up to V_{CCIO_SDM} or down to ground as required by the MSEL[2:0] setting for configuration scheme. You must also specify the configuration scheme on the **Configuration** page of the **Device and Pin Options** dialog box in the Intel Quartus Prime Software.

Figure 5. MSEL Pull-Up and Pull-Down Circuit Diagram

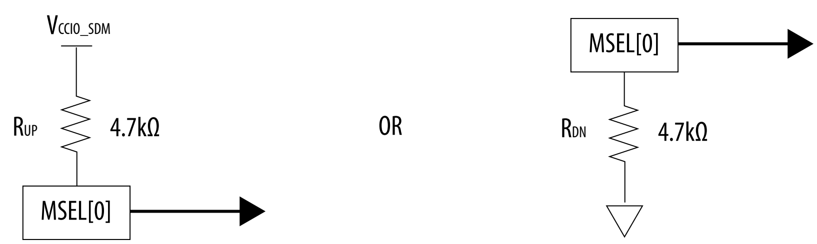


Table 4. MSEL Settings for Each Configuration Scheme of Intel Stratix 10 Devices

Configuration Scheme	MSEL[2:0]
Avalon-ST (x32)	000
Avalon-ST (x16)	101
Avalon-ST (x8)	110
AS (Fast mode – for CvP)[(2)]	001

continued...

Configuration Scheme	MSEL[2:0]
AS (Normal mode)	011
SD/MMC x4/x8	100
JTAG only(3)	111

Related Information

- Intel Stratix 10 GX and SX Device Family Pin Connection Guidelines
- POR Specifications in Intel Stratix 10 Device Datasheet

2.4.3. Device Configuration Pins

All configuration schemes use the same dedicated pins for the standard control signals shown in Intel Stratix 10 Configuration Timing Diagram on page 14.
There are no dedicated pins for the following signals:

- PR_REQUEST
- PR_ERROR
- PR_DONE
- CvP_CONFDONE
- SEU_ERROR
- DIRECT_TO_FACTORY

You can use the unused SDM I/O pins for CvP_CONFDONE, DIRECT_TO_FACTROY, and SEU_ERROR pins. You can only use GPIO for PR_REQUEST, PR_ERROR, and PR_DONE pins by specifying them in the Intel Quartus Prime software and connecting them to the Partial Reconfiguration External Configuration Controller Intel Stratix 10 FPGA IP.

Table 5. Intel Stratix 10 Device Configuration Pins

Configuration Function	Configuration Scheme	Direction	Powered by
TCK(4)	JTAG	Input	V_{CCIO_SDM}
TDI(4)	JTAG	Input	V_{CCIO_SDM}
TMS(4)	JTAG	Input	V_{CCIO_SDM}
TDO(4)	JTAG	Output	V_{CCIO_SDM}
nSTATUS	All schemes	Output	V_{CCIO_SDM}
nCONFIG	All schemes	Input	V_{CCIO_SDM}
MSEL[2:0](5)	All schemes	Input	V_{CCIO_SDM}

continued...

(2) If you use AS Fast mode and are not concerned about 100 ms PCIe linkup, you must still ramp the VCCIO_SDM supply within 18 ms. This ramp-up requirement ensures that the AS x4 device is within its operating voltage range when the Intel Stratix 10 device begins to access it.

(3) JTAG configuration works with any MSEL settings, unless disabled for security.

(4) The JTAG pins can access the HPS JTAG chain in Intel Stratix 10 SoC devices.

Configuration Function	Configuration Scheme	Direction	Powered by
CONF_DONE(6)	All schemes	Output	V_{CCIO_SDM}
INIT_DONE(7)	All schemes	Output	V_{CCIO_SDM}
OSC_CLK_1	All schemes	Input	V_{CCIO_SDM}
AS_nCSO[3:0]	AS	Output	V_{CCIO_SDM}
AS_DATA[3:0]	AS	Bidirectional	V_{CCIO_SDM}
AS_CLK	AS	Output	V_{CCIO_SDM}
AVST_READY	Avalon-ST x8/x16/32	Output	V_{CCIO_SDM}
AVSTx8_DATA[7:0]	Avalon-ST x8	Input	V_{CCIO_SDM}
AVSTx8_VALID	Avalon-ST x8	Input	V_{CCIO_SDM}
AVSTx8_CLK	Avalon-ST x8	Input	V_{CCIO_SDM}
AVST_DATA[31:0](8)	Avalon-ST x16/x32	Input	V_{CCIO}
AVST_VALID(8)	Avalon-ST x16/x32	Input	V_{CCIO}
AVST_CLK(8)	Avalon-ST x16/x32	Input	V_{CCIO}
SDMMC_CFG_CMD	SD/MMC	Output	V_{CCIO_SDM}
SDMMC_CFG_DATA[7:0]	SD/MMC	Bidirectional	V_{CCIO_SDM}
SDMMC_CFG_CCLK	SD/MMC	Output	V_{CCIO_SDM}

2.4.3.1. Configuration Pins I/O Standard and Drive Strength

Table 6. Intel Stratix 10 Configuration Pins I/O Standard and Drive Strength

Configuration Pin Function	Direction	I/O Standard	Drive Strength (mA)
TDO	Output	1.8V LVCMOS	8
TMS	Input	Schmitt Trigger Input	—
TCK	Input	Schmitt Trigger Input	—
TDI	Input	Schmitt Trigger Input	—
nSTATUS	Output	1.8V LVCMOS	8
OSC_CLK_1	Input	Schmitt Trigger Input	—

continued...

(5) The MSEL[2:0] pins are dual purpose. You can assign any unused MSEL[2:0] pin to other functions such as power management or non-dedicated configuration pins.

(6) You enable the CONF_DONE pin function in the Intel Quartus Prime Software. The Avalon-ST configuration scheme using the Parallel Flash Loader (PFL) II requires this pin.

(7) You enable the INIT_DONE pin function in the Intel Quartus Prime Software. This pin is optional for all configuration schemes.

(8) These are dual purpose configuration pins. You can use these pins as GPIOs in user mode.

Configuration Pin Function	Direction	I/O Standard	Drive Strength (mA)
nCONFIG	Input	Schmitt Trigger Input	—
SDM_IO[16:0]	I/O	Schmitt Trigger Input or 1.8V LVCMOS	8
All other configuration pins	I/O	Schmitt Trigger Input or 1.8V LVCMOS	8

Unused SDM Pins

You can specify other functions on unused SDM pins in the Intel Quartus Prime software.

Table 7. Additional Configuration Pins

Note: To avoid false signaling indicating successful configuration, Intel recommends that you include an external weak pull-down resistor for CONF_DONE and INIT_DONE pins.

Pin Function	Possible Settings	Recommended Settings	Functional Description
CONF_DONE	• **SDM_IO5**(9) • **SDM_IO16**	**SDM_IO16**	Allows you to monitor if device configuration is completed. During power-up, the SDM boot-up, and configuration stages, the pin is pulled low. Upon successful configuration, the pin is driven high by the Intel Stratix 10 device.
INIT_DONE	• **SDM_IO0** • **SDM_IO16** • **SDM_IO5**(10)	**SDM_IO0**	Allows you to monitor if device initialization is completed. During power-up, the SDM boot-up, configuration, and initialization stages, the pin is pulled low. Upon successful initialization, the pin is driven high by the Intel Stratix 10 device.

SDM pins are also available for the SmartVID power management feature for -V devices. You must also set the correct Power Management Bus (PMBus) settings when using the SmartVID feature. Refer to the *Intel Stratix 10 Power Management User Guide* for more information about the pin assignments and PMBus setting.

Related Information

Intel Stratix 10 Power Management User Guide

2.4.4. Setting Additional Configuration Pins

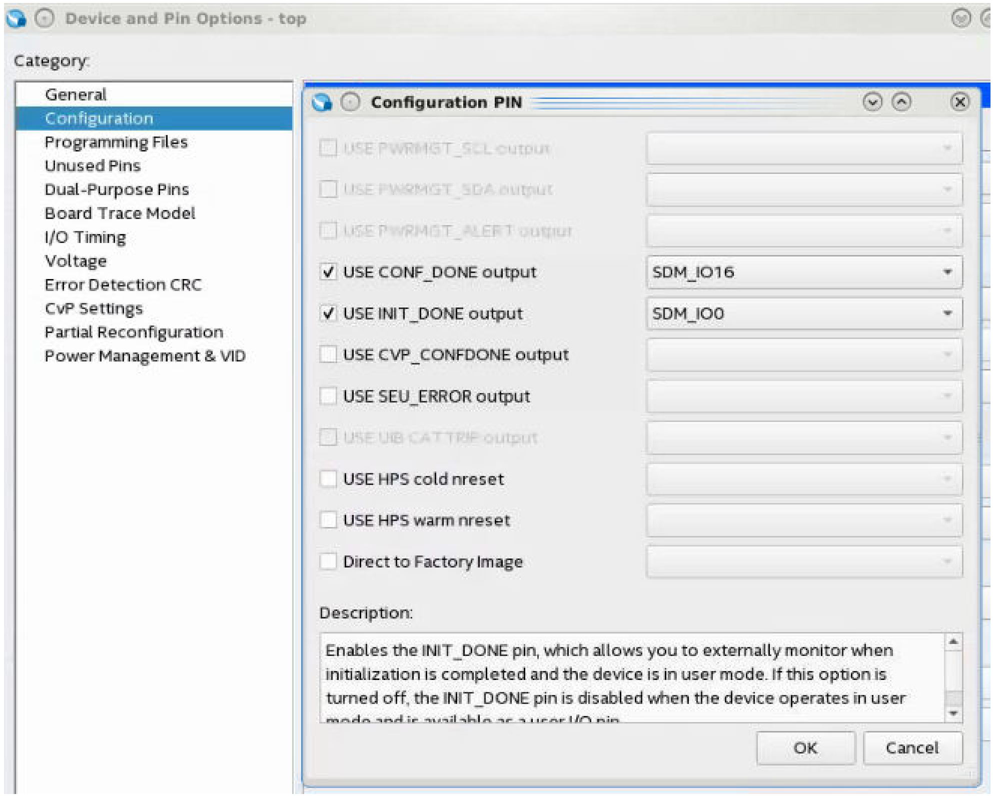
You enable and assign the SDM pins for CONF_DONE and INIT_DONE functions in the Intel Quartus Prime software.

Complete the following steps to assign additional configuration pins:

1. On the **Assignments** menu, click **Device**.
2. In the **Device and Pin Options** dialog box, select the **Configuration** category and click **Configuration Pins Options**.
3. In the **Configuration Pin** window, enable and assign the configuration pin that you want to enable.

(9) You can set CONF_DONE to SDM_IO5 when using Avalon-ST x8 and x32 schemes only.

(10) You can set INIT_DONE to SDM_IO5 when using Avalon-ST x8 and x32 schemes only.



4. Click **OK** to confirm and close the **Configuration Pin** dialog box.

2.4.5. Enabling Dual-Purpose Pins

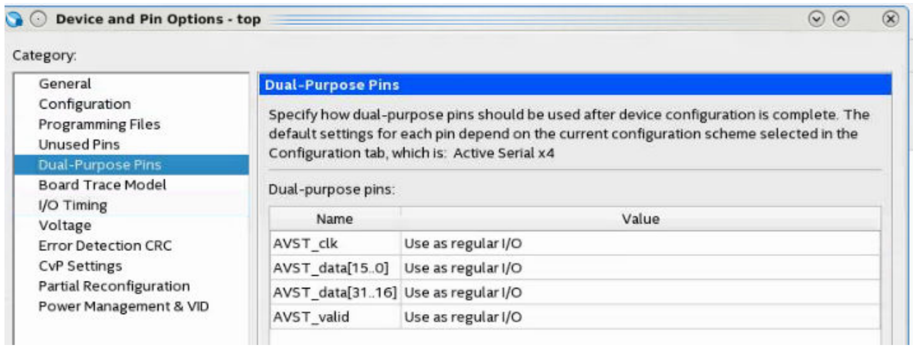
AVST_CLK, AVST_DATA[15:0], AVST_DATA[31:16], and AVST_VALID are dual-purpose pins. Once the device enters user mode these pins can function either as GPIOs or as tri-state inputs.

If you use these pins as GPIOs, make the following assignments:

- Set V_{CCIO} of the I/O bank at 1.8V
- Assign the 1.8V I/O standard to these pins

Complete the following steps to assign these settings to the dual-purpose pins:

1. On the **Assignments** menu, click **Device**.
2. In the **Device and Pin Options**, select the **Dual-Purpose Pins** category.
3. In the **Dual-purpose pins** table, set the pin functionality in the **Value** column.



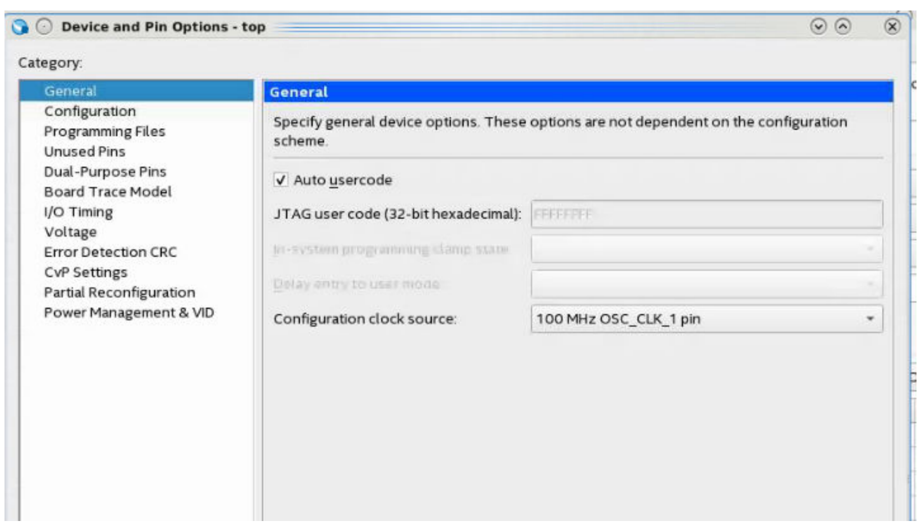
4. Click **OK** to confirm and close the **Device and Pin Options**.

2.5. Setting Configuration Clock Source

You must specify the configuration clock source by selecting either the internal oscillator or OSC_CLK_1 with the supported frequency. By default, the SDM uses the internal oscillator for device configuration. Specify an OSC_CLK_1 clock source for the fastest configuration time.

Complete the following steps to select the configuration clock source:

1. Specify an OSC_CLK_1 clock source for the fastest configuration time. On the **Assignments** menu, click **Device**.
2. In the **Device and Pin Options** select the **General** category.
3. Specify the configuration clock source from the **Configuration clock source** drop down menu.



4. Click **OK** to confirm and close the **Device and Pin Options**.

Related Information

OSC_CLK_1 Clock Input on page 24

2.6. Configuration Clocks

2.6.1. OSC_CLK_1 Clock Input

When you drive OSC_CLK_1 input clock with an external clock source and enable OSC_CLK_1 in the Intel Quartus Prime software, the device loads the majority of the configuration bitstream at 250 MHz. Intel Stratix 10 devices include an internal oscillator in addition to OSC_CLK_1 which run the configuration process at a frequency between 170-230 MHz. Intel Stratix 10 devices always use this internal oscillator to load the first section of the bitstream (approximately 200 kilobyte (KB). The SDM can use either clock source for the remainder of device configuration. If you use the internal oscillator, can you leave the OSC_CLK_1 unconnected.

Note: Device configuration may fail under the following conditions when you select the OSC_CLK_1 as the clock source for configuration:

- You fail to drive the OSC_CLK_1 pin.
- You drive the OSC_CLK_1 pin at an incorrect frequency. Select one of the following input reference clock frequencies to drive the OSC_CLK_1 pin:
  - 25
  - 100
  - 125

The Intel Stratix 10 device multiplies the OSC_CLK_1 source clock frequency to generate a 250 MHz clock for configuration. Using an OSC_CLK_1 source enables the fastest possible configuration. Refer to *Setting Configuration Clock Source* for instructions on completing this task.

Related Information

- Intel Stratix 10 L- and H-Tile Transceiver PHY User Guide
- Intel Stratix 10 E-Tile Transceiver PHY User Guide
- Intel Stratix 10 External Memory Interfaces IP User Guide
- Setting Configuration Clock Source on page 23

2.7. Configuration and Programming Files

Intel Stratix 10 configuration and external flash programming involves multiple file types and tools.

Figure 6. Overview of Intel Quartus Prime Supported Files and Tools for Configuration and Programming

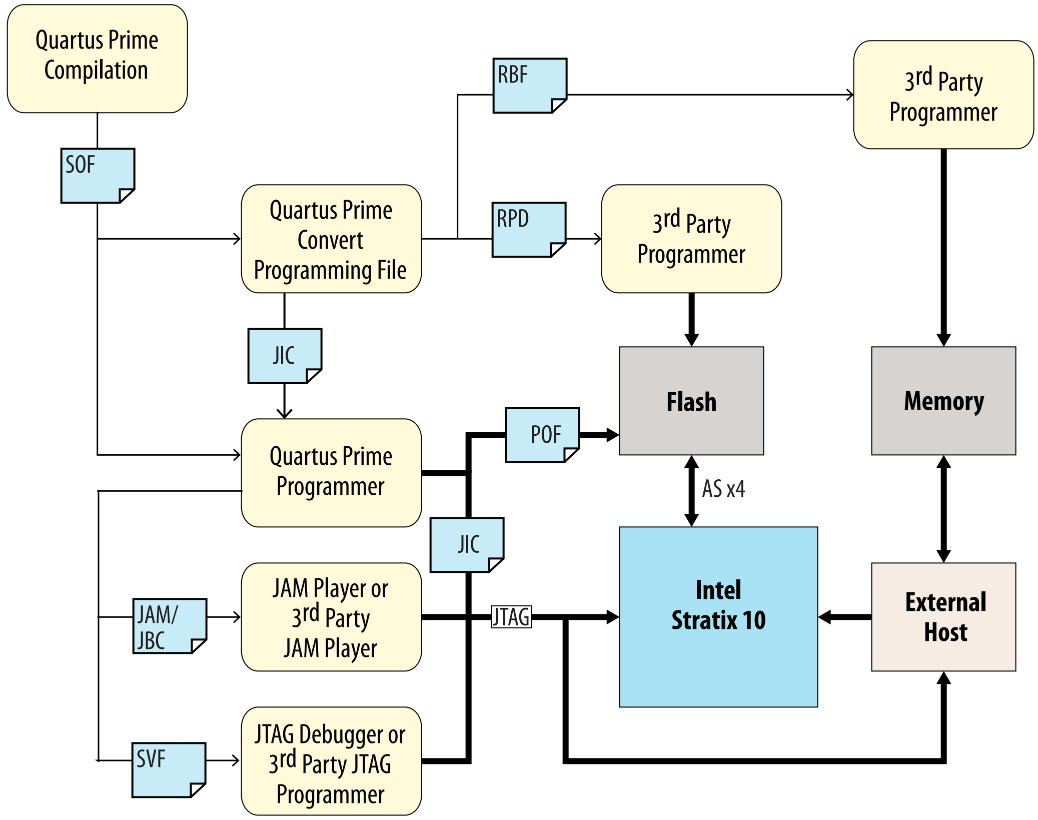


Table 8. Supported Programming and Configuration File Format

File Format	Description
SRAM Object File (`.sof`/SOF)	Configuration file for JTAG configuration
Raw Binary File (`.rbf`/RBF)	Configuration file for use with a third-party data source, CvP, partial reconfiguration, or HPS data source
Programming Object File (`.pof`/POF)	Serial flash and external flash programming file for AS and Avalon-ST configuration using Intel Quartus Prime Programmer
JTAG Indirect Configuration File (`.jic`/JIC)	Serial flash programming file for AS configuration using Intel Quartus Prime Programmer
Raw Programming Data File (`.rpd`/RPD)	Serial flash programming file for AS configuration using third-party programmer

continued...

File Format	Description
JAM Standard Test and Programming Language Format (.jam/JAM)	Configuration file for third-party JTAG host
JAM Byte Code (.jbc/JBC)	
Serial Vector Format (.svf/SVF)	

Related Information

- Can I use 3rd party QSPI flash devices for Active Serial configuration of Intel Stratix 10 devices?
- Using the Command-Line Jam STAPL Solution for Device Programming
- Intel FPGA IP for Configuration - Support Center

3. Intel Stratix 10 Configuration Schemes

3.1. Avalon-ST Configuration

The Avalon-ST configuration scheme is new in Intel Stratix 10 devices. It replaces the FPP mode available in earlier device families. The Avalon-ST configuration scheme is passive. Avalon-ST is the fastest configuration scheme for Intel Stratix 10 devices. This scheme uses an external host, such as a microprocessor, MAX® II, MAX V, or Intel MAX 10 device to drive configuration. The external host controls the transfer of configuration data from an external storage such as flash memory to the FPGA. The design that controls the configuration process resides in the external host. You can use the PFL II IP core with a MAX II, MAX V, or Intel MAX 10 device as the host to read configuration data from the flash memory device and configure the Intel Stratix 10 device.

Table 9. Avalon-ST Configuration Data Width, Clock Rates, and Data Rates

Protocol	Data Width (bits)	Max Clock Rate	Max Data Rate	MSEL[2:0]
Avalon-ST	32	125 MHz	4000 Mbps	000
	16	125 MHz	2000 Mbps	101
	8	125 MHz	1000 Mbps	110

Refer to the *Intel Stratix 10 Datasheet* for configuration timing estimates.

The Avalon-ST configuration scheme supports the following configuration methods:

- CPLD with PFL II and common flash interface (CFI) flash memory
- External host, typically a microprocessor, with any external memory

Note: You can use the Intel PFL II IP core as the configuration host. If you use a third-party microprocessor, refer to the *Avalon Streaming Interfaces* in the *Avalon Interface Specifications* for protocol details.

Related Information

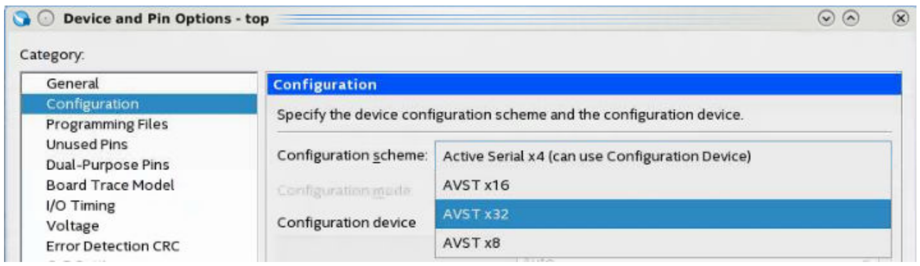
- Avalon Interface Specifications
- Intel Stratix 10 Device Features
  For a list of Intel Stratix 10 device features that are planned for future releases.

3.1.1. Enabling Avalon-ST Device Configuration

You enable the Avalon-ST device configuration scheme in the Intel Quartus Prime software.

Complete the following steps to specify an Avalon-ST interface for device configuration.

1. On the **Assignments** menu, click **Device**.
2. In the **Device and Pin Options** dialog box, select the **Configuration** category.
3. In the **Configuration** window, in the **Configuration scheme** dropdown list, select the appropriate Avalon-ST bus width.

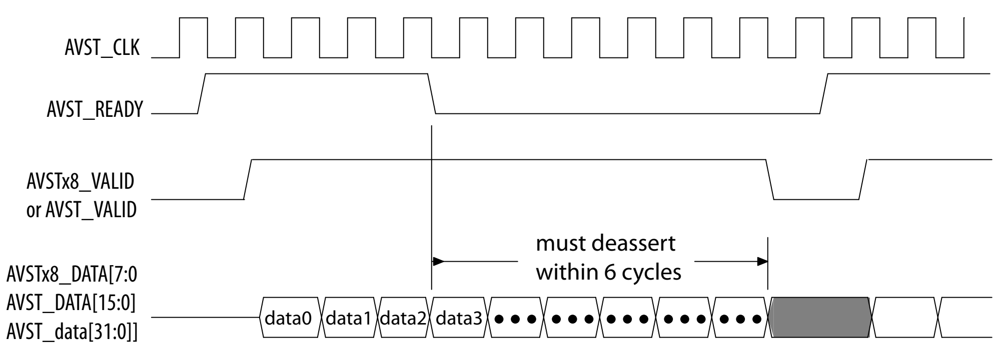


4. Click **OK** to confirm and close the **Device and Pin Options** dialog box.

3.1.2. Avalon-ST Configuration Timing

Before beginning configuration, trigger device cleaning by toggling the `nCONFIG` pin from high to low to high. These `nCONFIG` transitions also return the device to the configuration state.

Figure 7. Avalon-ST Bus Timing Waveform



The configuration files for Intel Stratix 10 devices can be highly compressed. During configuration, the decompression of the bit stream inside the device requires the host to pause before sending more data. The Intel Stratix 10 device asserts the AVST_READY signal when the device is ready to accept data. The AVST_READY signal is only valid when the nSTATUS pin is high. In addition, the host must handle backpressure by monitoring the AVST_READY signal and may assert `AVST_VALID` signal any time after the assertion of AVST_READY signal. The host must monitor the AVST_READY signal throughout the configuration.

The AVST_READY signal sent by the Intel Stratix 10 device to the host is not synchronized with the AVSTx8_CLK or AVST_CLK. To configure the Intel Stratix 10 device successfully, the host must adhere to the following constraints:

- The host must drive no more than six data words after the deassertion of the AVST_READY signal including the delay incurred by the 2-stage register synchronizer.
- The host must synchronize the AVST_READY signal to the AVST_CLK signal using a 2-stage register synchronizer. Here is Register transfer level (RTL) example code for 2-stage register synchronizer:

```
always @(posedge avst_clk or negedge reset_n)
    begin
        if (~reset_n)
        begin
            fpga_avst_ready_reg1 <= 0;
            fpga_avst_ready_reg2 <= 0;
        else
            fpga_avst_ready_reg1 <= fpga_avst_ready;
            fpga_avst_ready_reg2 <= fpga_avst_ready_reg1;
        end
    end
```

  Where:

  — The AVST_CLK signal comes from either PFL II IP or your Avalon-ST controller logic.

  — `fpga_avst_ready` is the AVST_READY signal from the Intel Stratix 10 device

  — `fpga_avst_ready_reg2` signal is the AVST_READY signal that is synchronous to AVST_CLK.

You must properly constrain the AVST_CLK and AVST_DATA signals at the host. Perform timing analysis on both signals between the host and Intel Stratix 10 device to ensure the Avalon-ST configuration timing specifications are met. Refer to the *Avalon-ST Configuration Timing* section of the *Intel Stratix 10 Device Datasheet* for information about the timing specifications.

Note: The AVST_CLK signal must run continuously during configuration. The AVST_READY signal cannot assert unless the clock is running.

Optionally, you can monitor the CONF_DONE signal to indicate the flash has sent all the data to FPGA or to indicate the configuration process is complete.

If you use the PFL II IP core as the configuration host, you can use the Intel Quartus Prime software to store the binary configuration data to the flash memory through the PFL II IP core.

If you use the Avalon-ST Adapter IP core as part of the configuration host, set the **Ready Latency** value between 1- 6.

Avalon-ST x8 configuration scheme uses the SDM pins only. Avalon-ST x16 and x32 configuration scheme additionally use dual-purpose I/O pins that you can use as general-purpose IO pins after configuration.

Related Information

- Avalon-ST Configuration Timing in Intel Stratix 10 Device Datasheet
- Avalon Interface Specifications

3.1.3. Avalon-ST Single-Device Configuration

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* for additional information about individual pin usage and requirements.

Figure 8. Connections for Avalon-ST x8 Single-Device Configuration

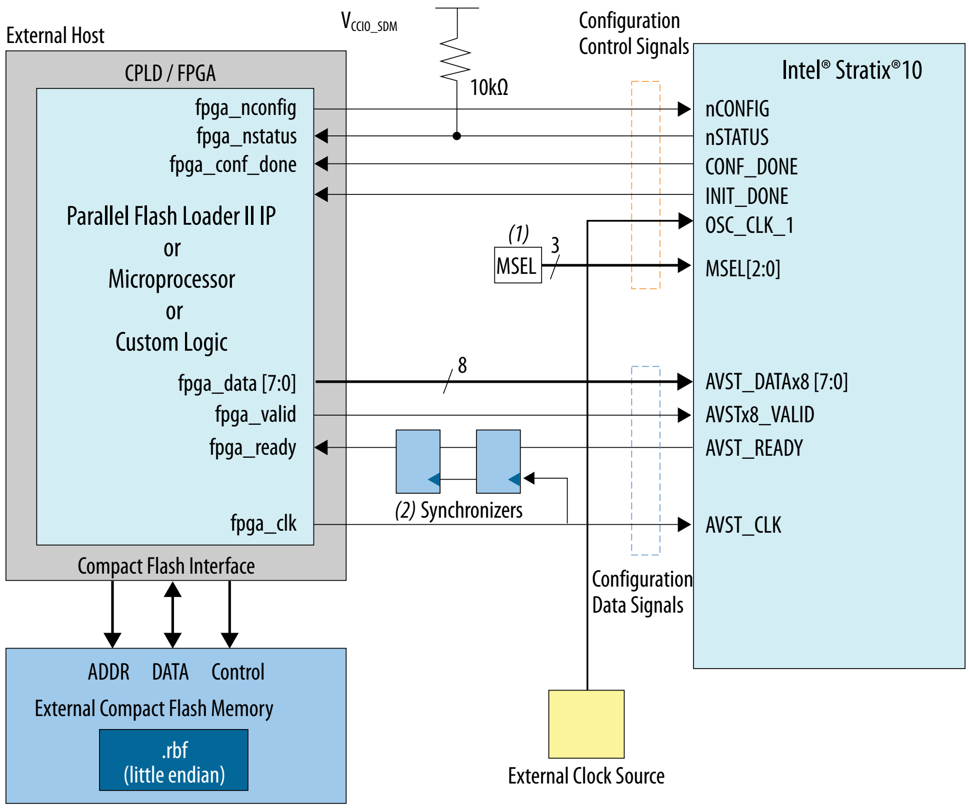


Figure 9. Connections for Avalon-ST x16 Single-Device Configuration

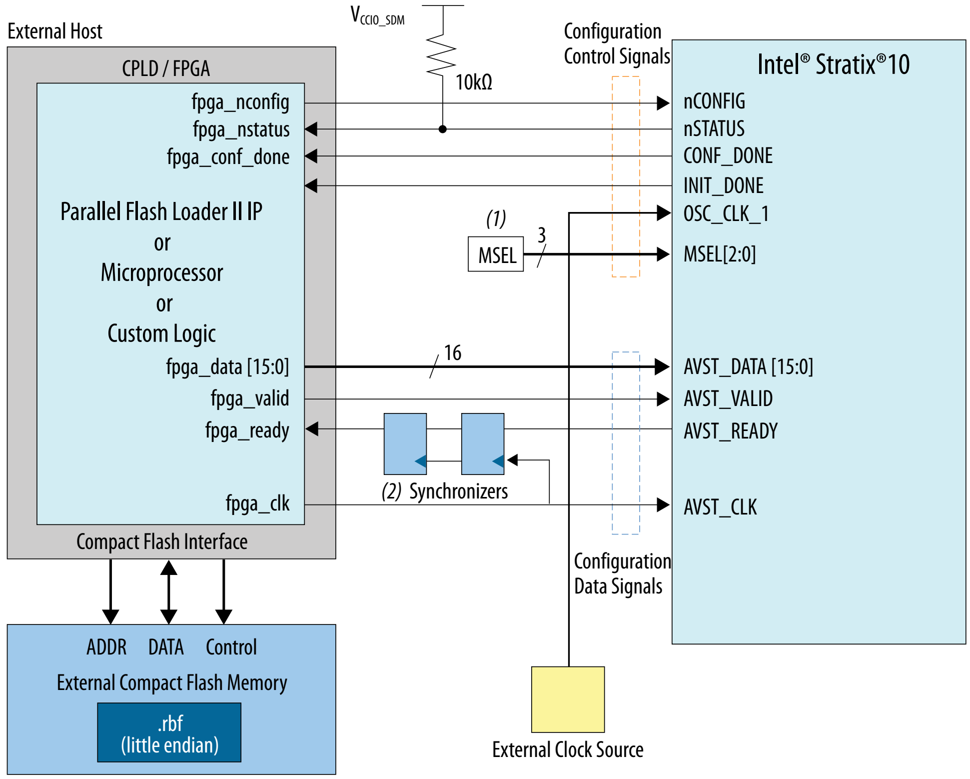
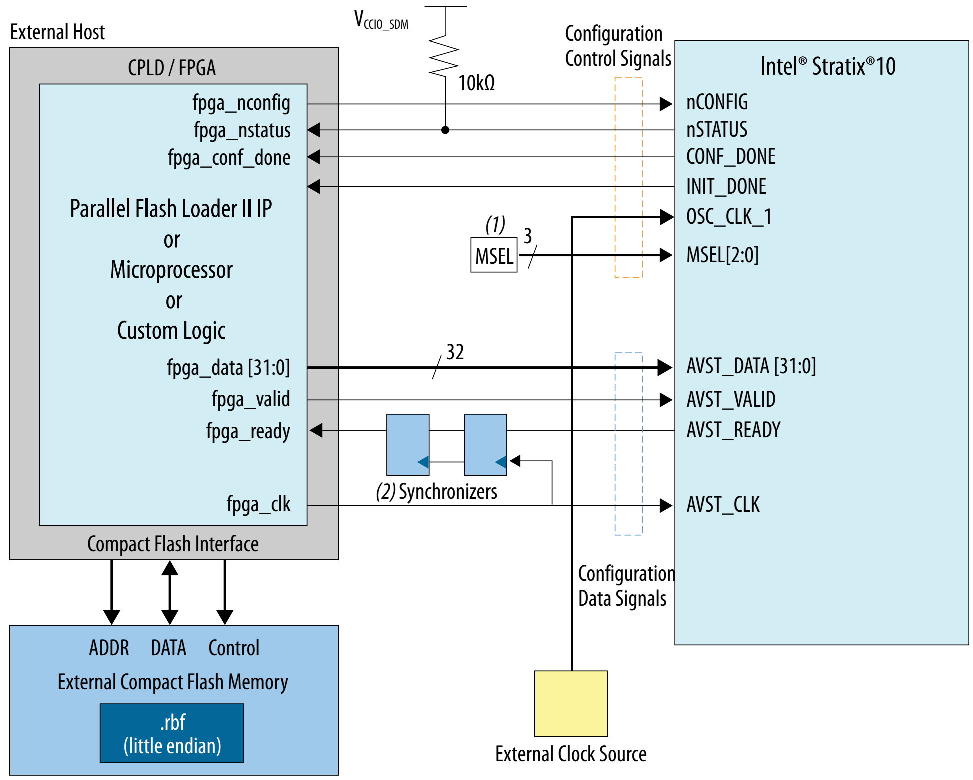


Figure 10. Connections for Avalon-ST x32 Single-Device Configuration



Note: The synchronizers shown in all three figures can be internal if the host is an FPGA or CPLD. If the host is a microprocessor, you must use discrete synchronizers.

Notes for Figure:

1. Refer to *MSEL Settings* for the correct resistor pull-up and pull-down values for all configuration schemes.
2. The synchronizers shown in all three figures can be internal if the host is an FPGA or CPLD. If the host is a microprocessor, you must use discrete synchronizers.

Related Information

- MSEL Settings on page 18
- Intel Stratix 10 Device Family Pin Connection Guidelines

3.1.4. RBF Configuration File Format

If you do not use the Parallel Flash Loader II Intel FPGA IP core to program the flash, you must generate the `.rbf` file.

The data in `.rbf` file are in little-endian format

Table 10. Writing 32-bit Data

For a x32 data bus , the first byte in the file is the least significant byte of the configuration double word, and the fourth byte is the most significant byte.

Double Word = 01EE1B02			
LSB: BYTE0 = 02	BYTE1 = 1B	BYTE2 = EE	MSB: BYTE3 = 01
D[7:0]	D[15:8]	D[23:16]	D[31:24]
0000 0010	0001 1011	1110 1110	0000 0001

Table 11. Writing 16-bit Data

For a x16 data bus, the first byte in the file is the least significant byte of the configuration word, and the second byte is the most significant byte of the configuration word.

WORD0 = 1B02		WORD1 = 01EE	
LSB: BYTE0 = 02	MSB: BYTE1 = 1B	LSB: BYTE2 = EE	MSB: BYTE3 = 01
D[7:0]	D[15:8]	D[7:0]	D[15:8]
0000 0010	0001 1011	1110 1110	0000 0001

3.1.5. Debugging Guidelines for the Avalon-ST Configuration Scheme

the Avalon-ST configuration scheme replaces the previously available FPP modes. This configuration scheme retains similar functionality and performance. Here are the important differences:

- The Avalon-ST configuration scheme requires you to monitor the flow control signal, AVST_READY. The AVST_READY signal indicates if the device can receive configuration data.
- The AVST_CLK and AVSTx8_CLK clock signals cannot pauses when configuration data is not being transferred. Data is not transferred when AVST_READY and AVST_VALID are low. The AVST_CLK and AVSTx8_CLK clock signals must run continuously until CONF_DONE asserts.

Debugging Suggestions

Here are some debugging tips:

- Only assert AVST_VALID any time after AVST_READY asserts.
- Only assert AVST_VALID when the data is valid.
- Ensure that the AVST_CLK clock signal are continuous until CONF_DONE asserts.
- If using x8 mode, ensure that you use the dedicated SDM_IO pins for this interface (clock, data, valid and ready).
- If using x16 or x32 mode, power the IO bank containing the x16 or x32 pins (3A) at 1.8V.
- Ensure you select the appropriate Avalon-ST configuration scheme in your Intel Quartus Prime Pro Edition project.
- Ensure the MSEL pins reflect this mode.

3.1.6. IP for Use with the Avalon-ST Configuration Scheme: Intel FPGA Parallel Flash Loader II IP Core

3.1.6.1. Functional Description

You can either program the CPLD and the flash memory concurrently or separately.

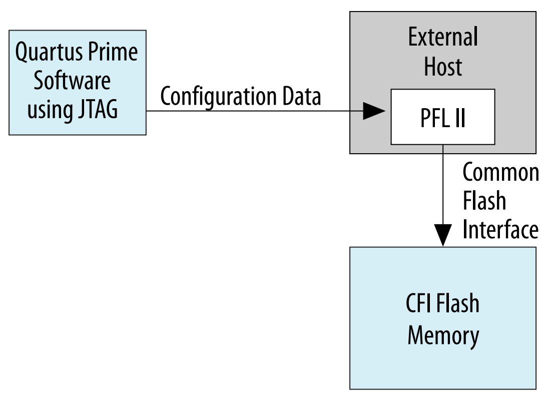
You can use the Parallel Flash Loader II Intel FPGA IP core (PFL II) with an external host, such as the MAX II, MAX V, or Intel MAX 10 devices to complete the following tasks:

- Program configuration data into a flash memory device using JTAG interface.
- Configure the Intel Stratix 10 device with the Avalon-ST configuration scheme from the flash memory device.

3.1.6.1.1. Programming CFI Flash

You can program the CFI flash using the PFL II IP core via the JTAG interface. Before you can program the CFI flash with configuration data, you must program the PFL II IP core into the host. You can only program with a `.pof` file and only use the Intel Quartus Prime Programmer to program the flash.

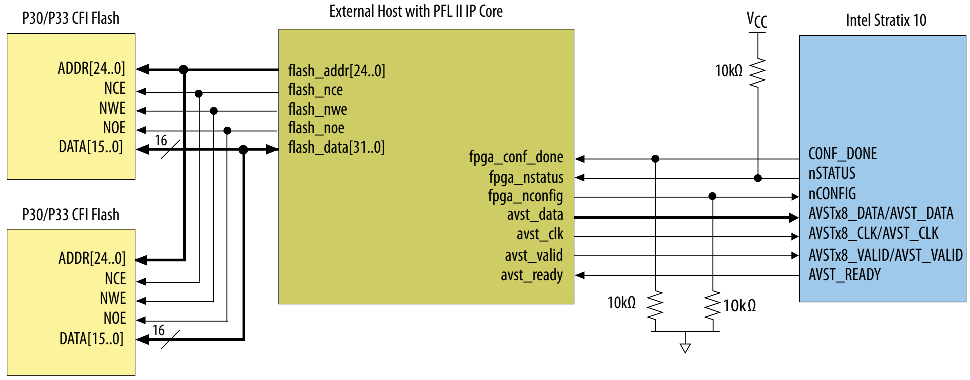
Figure 11. Programming the CFI Flash Memory with the JTAG Interface



The PFL II IP core supports dual P30 or P33 CFI flash memory devices in burst read mode to achieve faster configuration times. You can connect two identical P30 or P33 CFI flash memory devices to the host in parallel using the same data bus, clock, and control signals. During FPGA configuration, the AVST_CLK frequency is four times faster than the `flash_clk` frequency.

Figure 12. PFL II IP core with Dual P30 or P33 CFI Flash Memory Devices

The flash memory devices in the dual P30 or P33 CFI flash solution must have the same memory density from the same device family and manufacturer.



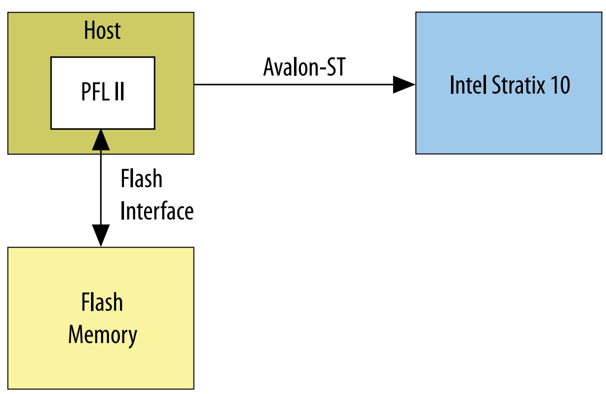
Related Information

Intel Stratix 10 GX FPGA Development Kit

3.1.6.1.2. Controlling Avalon-ST Configuration with PFL II IP Core

The PFL II IP core in the host determines when to start the configuration process, read the data from the flash memory device, and configure the Intel Stratix 10 using the Avalon-ST configuration scheme.

Figure 13. FPGA Configuration with Flash Memory Data



You can use the PFL II IP core to either program the flash memory devices, configure your FPGA, or both. To perform both functions, create separate PFL II functions if any of the following conditions apply to your design:

- You modify the flash data infrequently.
- You have JTAG or In-System Programming (ISP) access to the configuration host.
- You want to program the flash memory device with non-Intel FPGA data. For example, the flash memory device contains initialization storage for an ASSP. You can use the PFL II IP core to program the flash memory device for the following purposes:
  - — Write the initialization data
  - — Create your design source code to implement the read and initialization control with the host logic

3.1.6.1.3. Mapping PFL II IP Core and Flash Address

The address connections between the PFL II IP core and the flash memory device vary depending on the flash memory device vendor and data bus width.

Figure 14. Micron J3 Flash Memory in 8-Bit Mode

The address connection between the PFL II IP core and the flash memory device are the same.

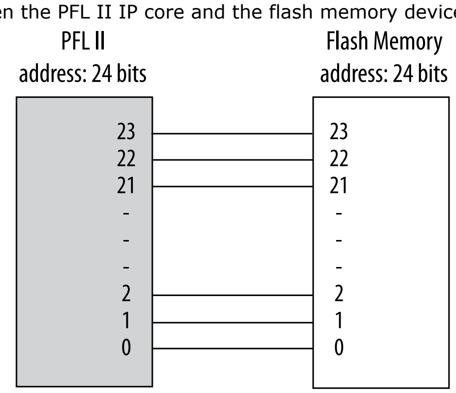


Figure 15. Micron J3, P30, and P33 Flash Memories in 16-Bit Mode

The flash memory addresses in Micron J3, P30, and P33 16-bit flash memory shift one bit down in comparison with the flash addresses in PFL II IP core. The flash address in the Micron J3, P30, and P33 flash memory starts from bit 1 instead of bit 0.

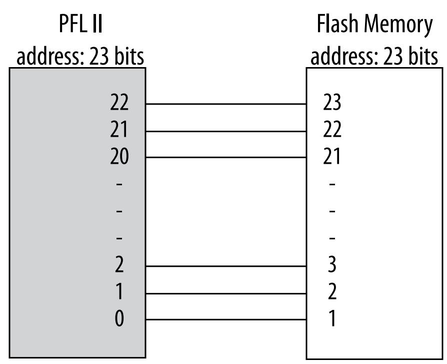


Figure 16. Cypress and Micron M28, M29 Flash Memory in 8-Bit Mode

The flash memory addresses in Cypress 8-bit flash shifts one bit up. Address bit 0 of the PFL II IP core connects to data pin D15 of the flash memory.

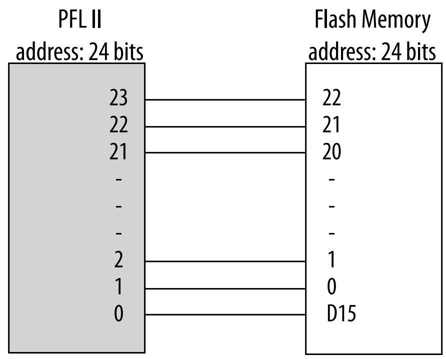
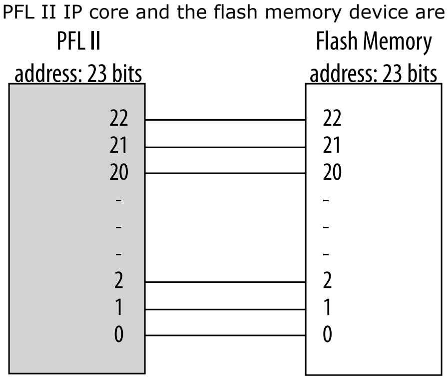


Figure 17. Cypress and Micron M28, M29 Flash Memory in 16-Bit Mode

The address bit numbers in the PFL II IP core and the flash memory device are the same.



3.1.6.1.4. Implementing Page in the Flash .pof

The PFL II IP core stores configuration data in a maximum of eight pages in a flash memory block. Each page holds the configuration data for a single FPGA chain.

The total number of pages and the size of each page depends on the density of the flash. These pages allow you to store designs for different FPGA chains or different designs for the same FPGA chain in different pages.

Use the generated `.sof` files to create a flash memory device `.pof`. When converting these `.sof` files to a `.pof`, use the following address modes to determine the page address:

- Block mode—allows you to specify the start and end addresses for the page.
- Start mode—allows you to specify only the start address. You can locate the start address for each page on an 8-KB boundary. If the first valid start address is `0×000000`, the next valid start address is an increment of `0×2000`.
- Auto mode—allows the Intel Quartus Prime software to automatically determine the start address of the page. The Intel Quartus Prime software aligns the pages on a 128-KB boundary; for example, if the first valid start address is `0×000000`, the next valid start address is an increment of `0×20000`.

3.1.6.1.5. Storing Option Bits

The PFL II IP core requires you to allocate space in the flash memory device for option bits. The option bits sector contains information about the start address for each page, the .pof version used for flash programming, and the Page-Valid bits. You must specify the options bits sector address in the flash memory device when converting the .sof files to a .pof and creating a PFL II design.

Table 12. Option Bits Sector Format

Offset address 0x80 stores the .pof version required for programming flash memory. This .pof version applies to all eight pages of the configuration data. The PFL II IP core requires the .pof version to perform a successful FPGA configuration process.

Sector Offset	Value
0x00–0x03	Page 0 start address
0x04–0x07	Page 0 end address
0x08–0x0B	Page 1 start address
0x0C–0x0F	Page 1 end address
0x10–0x13	Page 2 start address
0x14–0x17	Page 2 end address
0x18–0x1B	Page 3 start address
0x1C–0x1F	Page 3 end address
0x20–0x23	Page 4 start address
0x24–0x27	Page 4 end address
0x28–0x2B	Page 5 start address
0x2C–0x2F	Page 5 end address
0x30–0x33	Page 6 start address
0x34–0x37	Page 6 end address
0x38–0x3B	Page 7 start address
0x3C–0x3F	Page 7 end address
0x40–0x7F	Reserved
0x80(11)	.pof version
0x81-0xFF	Reserved

The Intel Quartus Prime Convert Programming File tool generates the information for the .pof version when you convert the .sof files to .pof files.

The value for the .pof version for Intel Stratix 10 is 0x05.

Caution: Do not overwrite any information in the option bits sector to prevent the PFL II IP core from malfunctioning, and always store the option bits in unused addresses in the flash memory device.

(11) .pof version occupies only one byte in the option bits sector.

3.1.6.1.6. Restoring Option Bit Start and End Address

You can restore the start and end address that you specified for each of the SOF page when converting a `.sof` to `.pof` file from the 32-bit value of the sector offset address.

The value for bit `[31:0]` for the start address of a page consists from the following format. The value for bit `[31:0]` for the end address of a page represents the 32 bits addressable end address.

Table 13. Start Address Bit Content

Bit	Width	Description
31:11	21	Addressable start address
10:1	10	Reserved bits
0	1	Page valid bit • 0=Valid • 1=Error

Table 14. End Address Bit Content

Bit	Width	Description
31:0	32	Addressable end address

To restore the addresses:

- Start address—append 13 bits of `0` to the addressable start address
- End address—append 2 bits of 1 to the addressable end address

You have a converted a `.pof` file that has two-page address with the following values in the option bit sector offset:

Sector Offset	Value
0x00 – 0x03	0x00004000
0x04 – 0x07	0x00196E30
0x08 – 0x0B	0x001C0000
0x0C – 0x0F	0x00352E30

Page 0 start address = Bit[31:11] appends with 0000000000000

= 00000000000000010000000000000000

= 0x10000

Page 0 end address = 0x00196E30 appends with 2'b11

= 0001100101101110001100011

= 0x65B8C3

Page 1 start address = Bit[31:11] appends with 0000000000000

= 00000000000111000000000000000000000

= 0x700000

Page 1 end address = 0x00352E30 appends with 2'b11

= 00000000001101010010111000110000 11

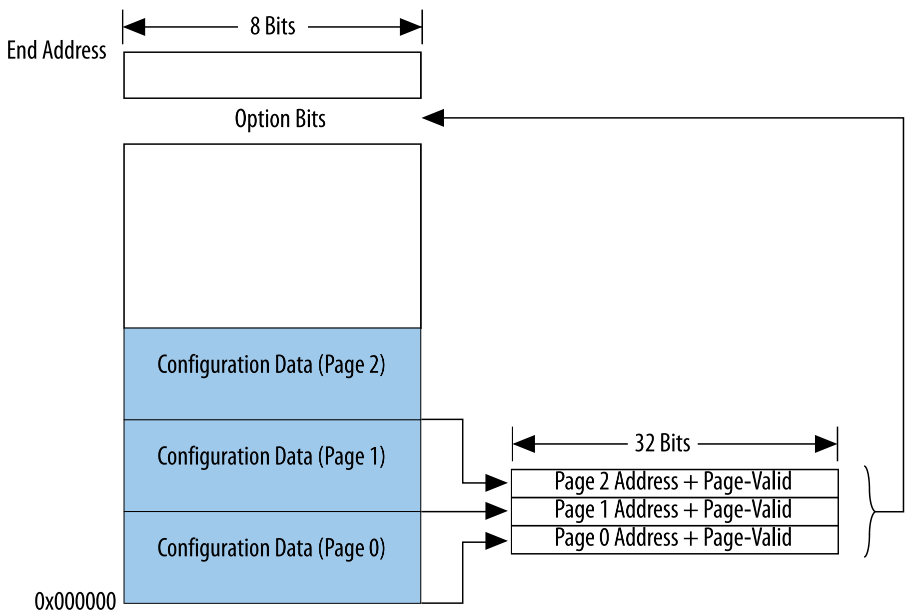
= 0xD4B8C3

The start and end address must be correlated with the start and end address for each page printed in the .map file.

3.1.6.1.7. Implementing Page Mode and Option Bits in the CFI Flash Memory Device

Figure 18. Implementing Page Mode and Option Bits in the CFI Flash Memory Device

- The end address depends on the density of the flash memory device. For the address range for devices with different densities, refer Byte Address Range table.
- You must specify the byte address for the option bits sector.



Use the parameter editor to set the option bits on the **FPGA Configuration** tab of Parallel Flash Loader II Intel FPGA IP.

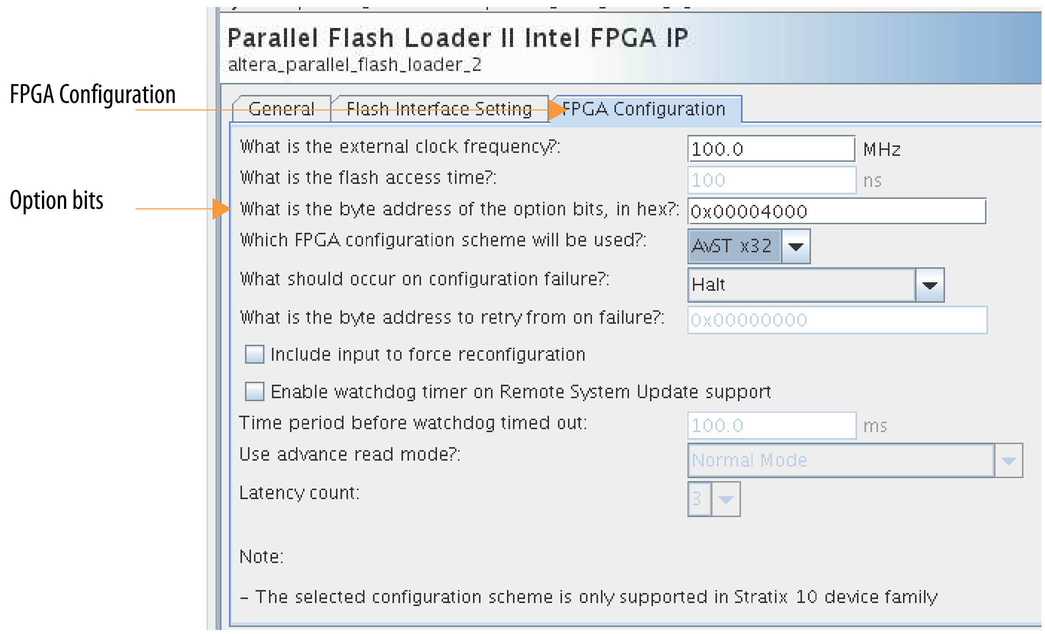


Figure 19. Page Start Address, End Address, and Page-Valid Bit Stored as Option Bits

Bits 0 to 12 for the page start address are set to zero and are not stored as option bits. The Page-Valid bits indicate whether each page is successfully programmed. The PFL II IP core programs the Page-Valid bits after successfully programming the pages.

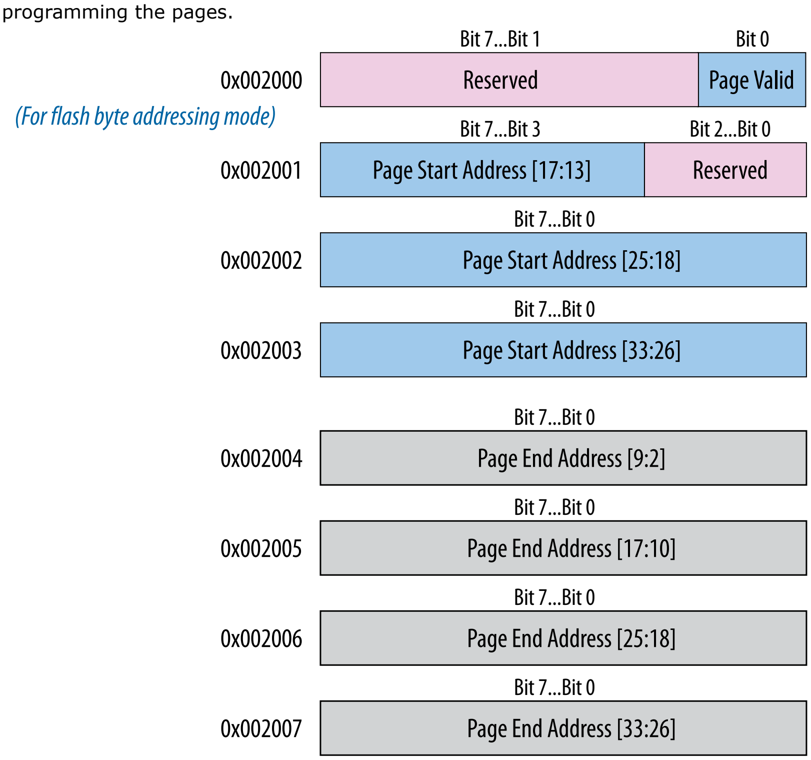


Table 15. Byte Address Range for CFI Flash Memory Devices with Different Densities

CFI Device (Megabit)	Address Range
8	0x0000000–0x00FFFFF
16	0x0000000–0x01FFFFF
32	0x0000000–0x03FFFFF
64	0x0000000–0x07FFFFF
128	0x0000000–0x0FFFFFF
256	0x0000000–0x1FFFFFF
512	0x0000000–0x3FFFFFF
1024	0x0000000–0x7FFFFFF

3.1.6.2. Using PFL II IP Core

3.1.6.2.1. Converting .sof to .pof File

To convert the `.sof` file to a `.pof`, follow these steps:

1. On the **File** menu, click **Convert Programming Files**.
2. For **Programming file type**, specify **Programmer Object File** (`.pof`) and name the file.
3. For **Configuration device**, select the CFI flash memory device with the correct density.

   For example, CFI_1Gb is a CFI device with 1-Gigabit (Mb) capacity.
4. For **Mode**, select the configuration scheme that matched to the `.sof` file. The available configuration modes are AvSTx8/AvSTx16/AvSTx32.
5. To add the configuration data, under **Input files to convert**, select **SOF Data**.
6. Click **Add File** and browse to the `.sof` files you want to add.

   You can place more than one `.sof` in the same page if you intend to configure a chain of FPGAs. The order of the `.sof` files must follow the order of the devices in the chain. If you want to store the data from other `.sof` files in a different page, click **Add SOF page**. Add the `.sof` files to the new page.
7. Select **SOF Data** and click **Properties** to set the page number and name. Under **Address scheme for selected pages**, select **Auto** to let the Intel Quartus Prime software automatically set the start address for that page. Select Block to specify the start and end addresses or select Start to specify the start address only and click OK.
8. You can also store Hexadecimal (Intel-Format) File (`.hex`) user data in the flash memory device:
   a. In the **Input files to convert** sub-window of the **Convert Programming Files**, select **Add Hex Data**.
   b. In the **Add Hex Data** dialog box, select either absolute or relative addressing mode.

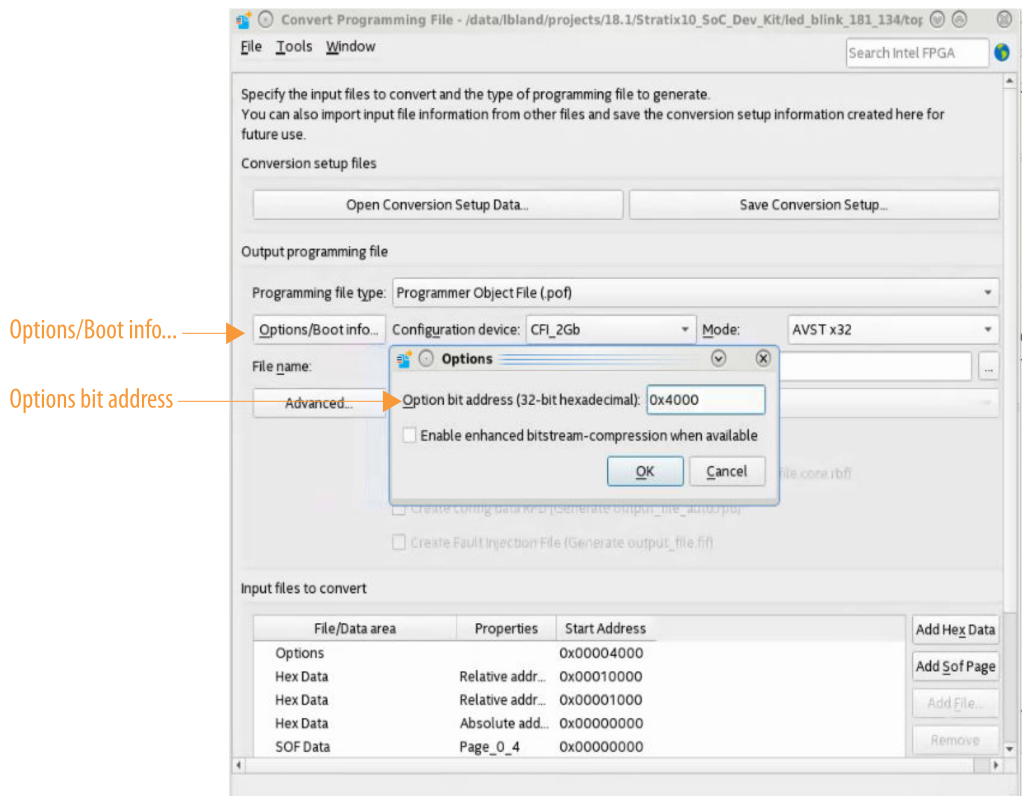
- If you select absolute addressing mode, the data in the `.hex` is programmed in the flash memory device at the same address location listed in the `.hex`.
- If you select relative addressing mode, specify a start address. The data in the `.hex` is programmed into the flash memory device with the specific start address, and the differences between the addresses are kept. If no address is specified, the software selects an address.

Note: You can also add other non-configuration data to the `.pof` by selecting the `.hex` that contains your data when creating the flash memory device `.pof`.

9. Click **Options/Boot info** to specify the start address to store the option bits.

   This start address must be identical to the address you specify when creating the PFL II IP core. Ensure that the option bits sector does not overlap with the configuration data pages and that the start address resides on an 8-KB boundary.

Figure 20. Convert Programming File - Options Bit Address



10. To generate programming files with the enhanced bitstream compression feature, turn on the **Enable enhanced bitstream-compression** when available in the **Options** dialog box and click **OK**.
11. Click **Generate** to create the `.pof`.

3.1.6.2.2. Creating Separate PFL II Functions

1. To create a PFL II instantiation, select **Flash Programming Only** mode.
2. Assign the pins appropriately.
3. Compile and generate a `.pof` for the flash memory device. Ensure that you tri-state all unused I/O pins.
4. To create another PFL II instantiation, select **Configuration Control Only mode**.
5. Instantiate this configuration controller into your production design.
6. Whenever you must program the flash memory device, program the CPLD with the flash memory device `.pof` and update the flash memory device contents.
7. Reprogram the host with the production design `.pof` that includes the configuration controller.

   Note: All unused pins are set to ground by default. When programming the configuration flash memory device through the host JTAG pins, you must tri-state the FPGA configuration pins common to the host and the configuration flash memory device. You can use the `pfl_flash_access_request` and `pfl_flash_access_granted` signals of the PFL II block to tri-state the correct FPGA configuration pins.

3.1.6.2.3. Programming CPLDs and Flash Memory Devices Concurrently

You can either program the CPLD and the flash memory concurrently or separately. To program concurrently, first program the CPLD, then the flash memory device. Follow these steps:

1. Open the **Programmer** and click **Add File** to add the `.pof` for the CPLD.
2. Right-click the **CPLD .pof** and click **Attach Flash Device**.
3. In the **Flash Device** menu, select the density of the flash memory device to be programmed.
4. Right-click the necessary flash memory device density and click **Change File**.
5. Select the `.pof` generated for the flash memory device. The `.pof` for the flash memory device is attached to the `.pof` of the CPLD.
6. Add other programming files if your chain has other devices.
7. Check all the boxes in the **Program/Configure** column for the new `.pof` and click **Start** to program the CPLD and flash memory device.

3.1.6.2.4. Programming CPLDs and Flash Memory Devices Separately

To program the CPLD and the flash memory devices separately, follow these steps:

1. Open the **Programmer** and click **Add File**.
2. In the **Select Programming File**, add the targeted `.pof`, and click **OK**.
3. Check the boxes under the **Program/Configure** column of the `.pof`.
4. Click **Start** to program the CPLD.
5. After the programming progress bar reaches 100%, click **Auto Detect**.

Send Feedback

For example, if you are using dual P30 or P33, the programmer window shows a dual P30 or P33 chain in your setup. Alternatively, you can add the flash memory device to the programmer manually. Right-click the CPLD `.pof` and click **Attach Flash Device**. In the **Select Flash Device** dialog box, select the device of your choice.

6. Right-click the necessary flash memory device density and click **Change File**.

   Note: You must select the density that is equivalent to the sum of the density of two CFI flash memory devices. For example, if you require two 512-Mb CFI flash memory devices, then select CFI 1 Gbit.

7. Select the `.pof` generated for the flash memory device. The `.pof` for the flash memory device is attached to the `.pof` of the CPLD.

8. Check the boxes under the **Program/Configure** column for the added `.pof` and click **Start** to program the flash memory devices.

   Note: The Programmer allows you to program, verify, erase, blank-check, or examine the configuration data page, the user data page, and the option bits sector separately, provided the CPLD contains the PFL II IP core. The programmer erases the flash memory device if you select the `.pof` of the flash memory device before programming. To prevent the Programmer from erasing other sectors in the flash memory device, select only the pages, .hex data, and option bits.

3.1.6.2.5. Defining New CFI Flash Memory Device

The PFL II IP core supports Intel-compatible and AMD-compatible flash memory devices. In addition to the supported flash memory devices, you can define the new Intel- or AMD-compatible CFI flash memory device in the PFL II-supported flash database using the Define new CFI flash memory device feature.

To add a new CFI flash memory device to the database or update a CFI flash memory in the database, follow these steps:

1. In the Programmer window, on the Edit menu, select **Define New CFI Flash Device**. The **Define CFI Flash Device** window appears. The following table lists the three functions available in the Define CFI Flash Device window.

Table 16. Functions of the Define CFI Flash Device Feature

Function	Description
New	Add new Intel- or AMD-compatible CFI flash memory device into the PFL II-supported flash database.
Edit	Edit the parameters of the newly added Intel- or AMD-compatible CFI flash memory device in the PFL II-supported flash database.
Remove	Remove the newly added Intel- or AMD-compatible CFI flash memory device from the PFL II-supported flash database.

2. To add a new CFI flash memory device or edit the parameters of the newly added CFI flash memory device, select **New** or **Edit**. The **New CFI Flash Device** dialog box appears.

3. In the **New CFI Flash Device** dialog box, specify or update the parameters of the new flash memory device. You can obtain the values for these parameters from the datasheet of the flash memory device manufacturer.

Table 17. Parameter Settings for New CFI Flash Device

Parameter	Description
CFI flash device name	Define the CFI flash name
CFI flash device ID	Specify the CFI flash identifier code
CFI flash manufacturer ID	Specify the CFI flash manufacturer identification number
CFI flash extended device ID	Specify the CFI flash extended device identifier, only applicable for AMD-compatible CFI flash memory device
Flash device is Intel compatible	Turn on the option if the CFI flash is Intel compatible
Typical word programming time	Typical word programming time value in μs unit
Maximum word programming time	Maximum word programming time value in μs unit
Typical buffer programming time	Typical buffer programming time value in μs unit
Maximum buffer programming time	Maximum buffer programming time value in μs unit

Note: You must specify either the word programming time parameters, buffer programming time parameters, or both. Do not leave both programming time parameters with the default value of zero.

4. Click **OK** to save the parameter settings.
5. After you add, update, or remove the new CFI flash memory device, click **OK**.

The Windows registry stores user flash information. Consequently, you must have system administrator privileges to store the parameters in the **Define New CFI Flash Device** window in the Intel Quartus Prime Pro Edition Programmer.

3.1.6.3. Parameters

Table 18. PFL II General Parameters

Options	Value	Description
Operating mode	• Flash Programming and FPGA Configuration • Flash Programming • FPGA Configuration	Specifies the operating mode of flash programming and FPGA configuration control in one IP core or separate these functions into individual blocks and functionality.
Targeted flash device	• CFI Parallel Flash	Specifies the flash memory device connected to the PFL II IP core.
Tri-state flash bus	• On • Off	Allows the PFL II IP core to tri-state all pins interfacing with the flash memory device when the PFL II IP core does not require access to the flash memory.

Table 19. PFL II Flash Interface Setting Parameters

Options	Value	Description
Number of flash devices used	• CFI Parallel Flash: 1–16	Specifies the number of flash memory devices connected to the PFL II IP core.
Largest flash density	• CFI Parallel Flash: 8 Mbit–2 Gbit	Specifies the density of the flash memory device to be programmed or used for FPGA configuration. If you have more than one flash memory device connected to the PFL II IP core, specify the largest flash memory device density.

continued...

Send Feedback

Options	Value	Description
		For dual P30/P33 CFI flash, select the density that is equivalent to the sum of the density of two flash memories. For example, if you use two 512-Mb CFI flashes, you must select **CFI 1 Gbit**.
Flash interface data width	CFI Parallel Flash: • 8 • 16 • 32	Specifies the flash data width in bits. The flash data width depends on the flash memory device you use. For multiple flash memory device support, the data width must be the same for all connected flash memory devices. Select the flash data width that is equivalent to the sum of the data width of two flash memories. For example, if you are targeting dual P30 or P33 solution, you must select **32 bits** because each CFI flash data width is 16 bits.
User control `flash_nreset` pin	• On • Off	Creates a `flash_nreset` pin in the PFL II IP core to connect to the reset pin of the flash memory device. A low signal resets the flash memory device. In burst mode, this pin is available by default. When using a Cypress GL flash memory, connect this pin to the RESET# pin of the flash memory.

Table 20. PFL II Flash Programming Parameters

Options	Value	Description
Flash programming IP optimization	• Area • Speed	Specifies the flash programming IP optimization. If you optimize the PFL II IP core for speed, the flash programming time is shorter, but the IP core uses more LEs. If you optimize the PFL II IP core for area, the IP core uses less LEs, but the flash programming time is longer.
FIFO size	—	Specifies the FIFO size if you select Speed for flash programming IP optimization. The PFL II IP core uses additional LEs to implement FIFO as temporary storage for programming data during flash programming. With a larger FIFO size, programming time is shorter.
Add Block-CRC verification acceleration support	• On • Off	Adds a block to accelerate verification.

Table 21. PFL II FPGA Configuration Parameters

Options	Value	Description
External clock frequency	—	Specifies the user-supplied clock frequency for the IP core to configure the FPGA. The clock frequency must not exceed two times the maximum clock (AVST_CLK) frequency acceptable by the FPGA for configuration. The PFL II IP core can divide the frequency of the input clock maximum by two.
Flash access time	—	Specifies the access time of the flash. You can get the maximum access time that a flash memory device requires from the flash datasheet. Intel recommends specifying a flash access time that is the same as or longer than the required time. For CFI parallel flash, the unit is in ns and for NAND flash, the unit is in us. NAND flash uses page instead of byte and requires more access time. This option is disabled for quad SPI flash.
Option bits byte address	—	Specifies the start address in which the option bits are stored in the flash memory. The start address must reside on an 8-KB boundary. See related for more information about option bits.

continued...

Options	Value	Description
FPGA configuration scheme	• Avalon-ST x8 • Avalon-ST x16 • Avalon-ST x32	Select the FPGA configuration scheme.
Configuration failure response options	• Halt • Retry same page • Retry from fixed address	Configuration behavior after configuration failure. • If you select **Halt**, the FPGA configuration stops completely after failure. • If you select **Retry same page**, after failure, the PFL II IP core reconfigures the FPGA with data from the same page of the failure. • If you select **Retry from fixed address**, the PFL II IP core reconfigures the FPGA with data from a fixed address in the next option field after failure.
Byte address to retry from on configuration failure	—	If you select **Retry from fixed address** for configuration failure option, this option specifies the flash address for the PFL II IP core to read from the reconfiguration for a configuration failure.
Include input to force reconfiguration	• On • Off	Includes an optional reconfiguration input pin (`pfl_nreconfigure`) to enable reconfiguration of the FPGA.
Watchdog timer	• On • Off	Enables a watchdog timer for remote system upgrade support. Turning on this option enables the `pfl_reset_watchdog` input pin and `pfl_watchdog_error` output pin and specifies the period before the watchdog timer times out. This watchdog timer is a time counter which runs at the `pfl_clk frequency`.
Time period before the watchdog timer times out	—	Specifies the time out period of the watchdog timer. The default time out period is 100 ms
Use advance read mode	• Normal Mode • Intel Burst Mode (P30 or P33) • Cypress Page Mode (GL) • Micron Burst Mode (M58BW)	An option to improve the overall flash access time for the read process during the FPGA configuration. • Normal mode—Applicable for all flash memory • Intel Burst mode—Applicable for Micron P30 and P33 flash memory only. Reduces sequential read access time • Cypress page mode—Applicable for Cypress GL flash memory only • Micron burst mode—Applicable for Micron M58BW flash memory only For more information about the read-access modes of the flash memory device, refer to the respective flash memory data sheet.
Latency count	• 3 • 4 • 5	Specify the latency count for Intel Burst Read mode. Only available when you enable Intel Burst Mode.

3.1.6.4. Signals

Table 22. PFL II Signals

Pin	Type	Weak Pull-Up	Function
pfl_nreset	Input	—	Asynchronous reset for the PFL II IP core. Pull high to enable FPGA configuration. To prevent FPGA configuration, pull low when you do not use the PFL II IP core. This pin does not affect the flash programming functionality of the PFL II IP core.
pfl_flash_access_granted	Input	—	Used for system-level synchronization. This pin is driven by a processor or any arbitrator that controls access to the flash. This active-high pin is connected permanently high if you want the PFL II IP core to function as the flash master. Pulling the pfl_flash_access_granted pin low prevents the JTAG interface from accessing the flash and FPGA configuration.
pfl_clk	Input	—	User input clock for the device. Frequency must match the frequency specified in the IP core and must not be higher than the maximum DCLK frequency specified for the specific FPGA during configuration. These pins are not available for the flash programming option in the PFL II IP core.
fpga_pgm[]	Input	—	Determines the page for the configuration. These pins are not available for the flash programming option in the PFL II IP core.
fpga_conf_done	Input	10 kΩ Pull-Up Resistor	Connects to the CONF_DONE pin of the FPGA. The FPGA releases the pin high if the configuration is successful. During FPGA configuration, this pin remains low. These pins are not available for the flash programming option in the PFL II IP core.
fpga_nstatus	Input	10 kΩ Pull-Up Resistor	Connects to the nSTATUS pin of the FPGA. This pin must be released high before the FPGA configuration and must stay high throughout FPGA configuration. If a configuration error occurs, the FPGA pulls this pin low and the PFL II IP core stops reading the data from the flash memory device. These pins are not available for the flash programming option in the PFL II IP core.
pfl_nreconfigure	Input	—	A low signal at this pin initiates FPGA reconfiguration. You can reconnect this pin to a switch for more flexibility to set this input pin high or low to control FPGA reconfiguration. When FPGA reconfiguration is initiated, the fpga_nconfig pin is pulled low to reset the FPGA device. The pfl_clk. pin registers this signal. These pins are not available for the flash programming option in the PFL II IP core.
pfl_flash_access_request	Output	—	Used for system-level synchronization. When necessary, this pin connects to a processor or an arbitrator. The PFL II IP core drives this pin high when the JTAG interface accesses the flash or the PFL II IP core configures the FPGA. This output pin works in conjunction with the flash_noe and flash_nwe pins.
flash_addr[]	Output	—	Address inputs for memory addresses. The width of the address bus line depends on the density of the flash memory device and the width of the

continued...

Pin	Type	Weak Pull-Up	Function
			flash_data bus. The output of this pin depends on the setting of the unused pins if you did not select the PFL II interface tri-state option when the PFL II is not accessing the flash memory device.
flash_data[]	Input or Output (bidirectional pin)	—	Data bus to transmit or receive 8- or 16-bit data to or from the flash memory in parallel. The output of this pin depends on the setting of the unused pins if you did not select the PFL II interface tri-state option when the PFL II is not accessing the flash memory device. (12)
flash_nce[]	Output	—	Connects to the nCE pin of the flash memory device. A low signal enables the flash memory device. Use this pin for multiple flash memory device support. The flash_nce pin is connected to each nCE pin of all the connected flash memory devices. The width of this port depends on the number of flash memory devices in the chain.
flash_nwe	Output	—	Connects to the nWE pin of the flash memory device. A low signal enables write operation to the flash memory device.
flash_noe	Output	—	Connects to the nOE pin of the flash memory device. A low signal enables the outputs of the flash memory device during a read operation.
flash_clk	Output	—	Used for burst mode. Connects to the CLK input pin of the flash memory device. The active edges of CLK increment the flash memory device internal address counter. The flash_clk frequency is half of the pfl_clk frequency in burst mode for single CFI flash. In dual P30 or P33 CFI flash solution, the flash_clk frequency runs at a quarter of the pfl_clk frequency. Use this pin for burst mode only. Do not connect these pins from the flash memory device to the host if you are not using burst mode.
flash_nadv	Output	—	Used for burst mode. Connects to the address valid input pin of the flash memory device. Use this signal for latching the start address. Use this pin for burst mode only. Do not connect these pins from the flash memory device to the host if you are not using burst mode.
flash_nreset	Output	—	Connects to the reset pin of the flash memory device. A low signal resets the flash memory device.

continued...

(12) Intel recommends not inserting logic between the PFL II pins and the host I/O pins, especially on the flash_data and fpga_nconfig pins.

Pin	Type	Weak Pull-Up	Function
fpga_nconfig	Open Drain Output	10-kW Pull-Up Resistor	Connects to the nCONFIG pin of the FPGA. A low pulse resets the FPGA and initiates configuration. These pins are not available for the flash programming option in the PFL II IP core. (12)
pfl_reset_watchdog	Input	—	A switch signal to reset the watchdog timer before the watchdog timer times out. Hold the signal high or low for at least two clock cycles of the pfl_clk frequency to correctly reset the watchdog timer.
pfl_watchdog_error	Output	—	A high signal indicates an error to the watchdog timer.

Related Information

Avalon Interface Specifications

3.2. AS Configuration

In AS configuration schemes, the SDM block in the Intel Stratix 10 device controls the configuration process and interfaces. The serial flash configuration devices store the configuration data. During AS Configuration, the SDM first powers on with boot ROM. Then, the SDM loads the initial configuration firmware from AS x4 flash. After the configuration firmware loads, this firmware controls the remainder of the configuration process, including I/O configuration and FPGA core configuration. Designs including an HPS, can use the HPS to access serial flash memory after the initial configuration.

Note: The serial flash configuration device must be fully powered up at the same time or before ramping up V_{CCIO_SDM} of the Intel Stratix 10 device.

The AS configuration scheme supports AS x4 (4-bit data width) mode only.

Table 23. Intel Stratix 10 Configuration Data Width, Clock Rates, and Data Rates

Mode		Data Width (bits)	Max Clock Rate	Max Data Rate	MSEL[2:0]
Active	Active Serial (AS)	4	133 MHz	532 Mbps	Fast mode - 001 Normal mode - 011

Refer to the related information for more information about enabling other flash device support.

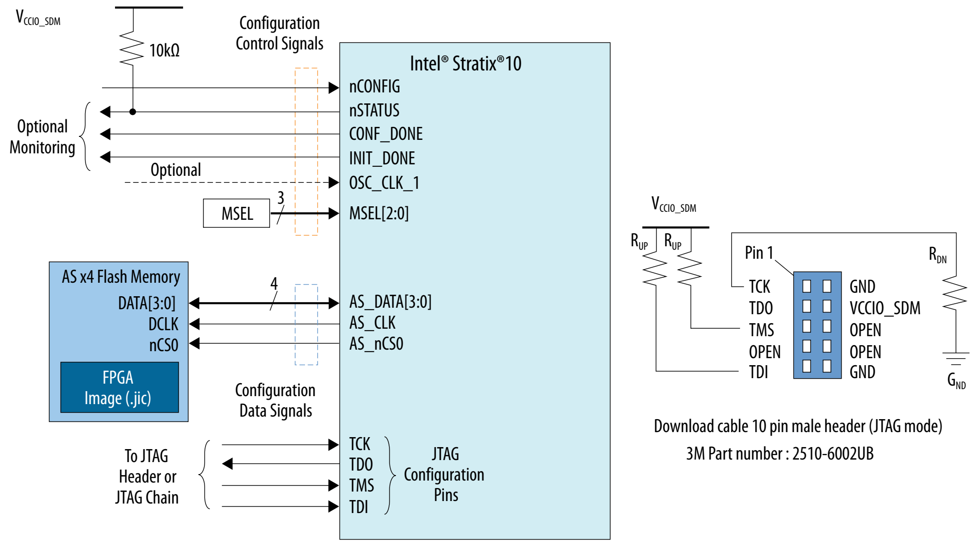
Related Information

- Can I use 3rd party QSPI flash devices for Active Serial configuration of Intel Stratix 10 devices?
- AS Configuration Timing in Intel Stratix 10 Devices

3.2.1. AS Single-Device Configuration

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* for additional information about individual pin usage and requirements.

Figure 21. Connections for AS x4 Single-Device Configuration



Related Information

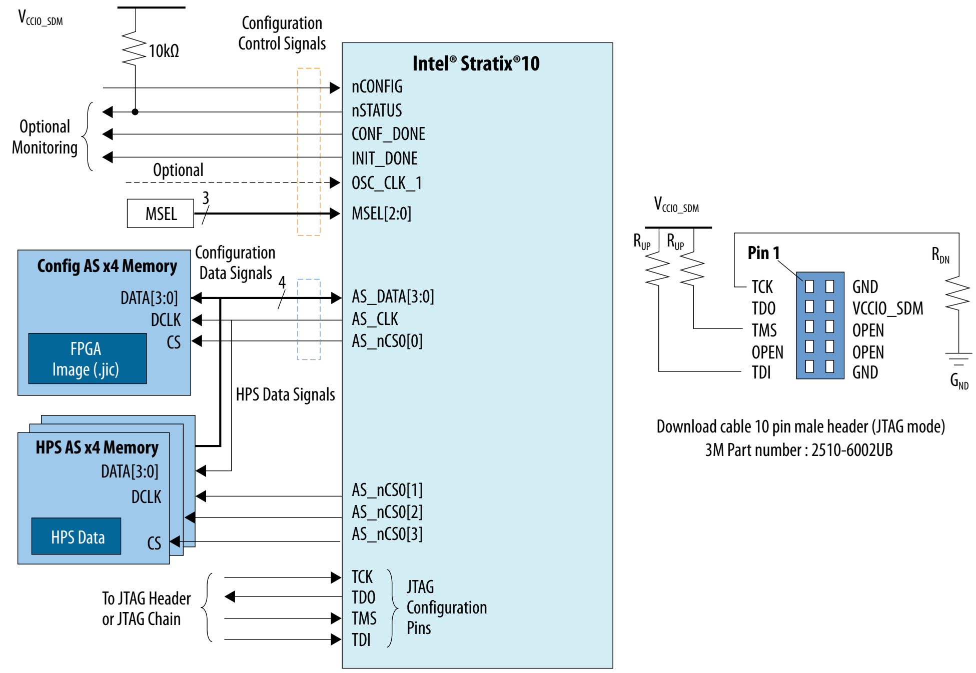
- MSEL Settings on page 18
- Intel Stratix 10 Device Family Pin Connection Guidelines

3.2.2. AS Using Multiple Serial Flash Devices

Intel Stratix 10 devices support one AS x4 flash memory device for AS configuration and up to three AS x4 flash memories for use with HPS data storage. The MSEL pins are dual-purpose and operate as MSEL only during POR state. After the FPGA device enters user mode, you can repurpose the MSEL pins as chip select pins. You must to ensure appropriate pin chip select pin connections to the configuration AS x4 flash memory and HPS AS x4 flash memory. Each flash device has a dedicated AS_nCS0 pin but shares other pins.

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* for additional information about individual pin usage and requirements.

Figure 22. Connection Setup for AS Configuration with Multiple Serial Flash Devices



To allow the JTAG interface to program the flash memory devices, set the MSEL pins to JTAG. When MSEL is set to JTAG, the SDM tristates the AS pins, AS_CLK, AS_DATA0-AS_DATA3, and AS_CS0-AS_CS3, when the device powers on.

Note: When using multiple flash devices, the clock frequency must be reduced. Refer to the *Intel Stratix 10 Device Datasheet* for more information.

Related Information

- MSEL Settings on page 18
- Intel Stratix 10 Device Datasheet (Core and HPS)
- Intel Stratix 10 Device Family Pin Connection Guidelines

3.2.3. AS Configuration Timing

Figure 23. AS Configuration Serial Output Timing Diagram

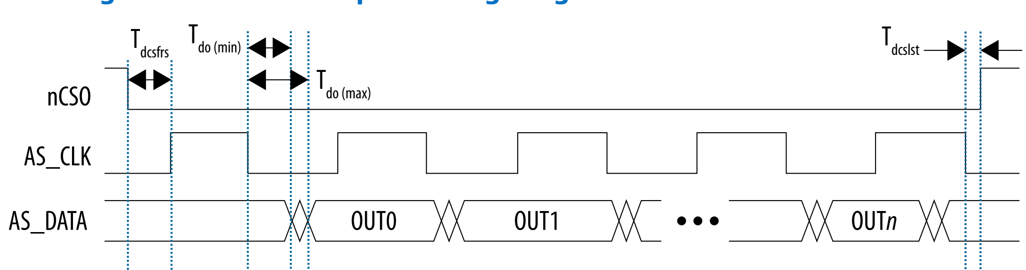
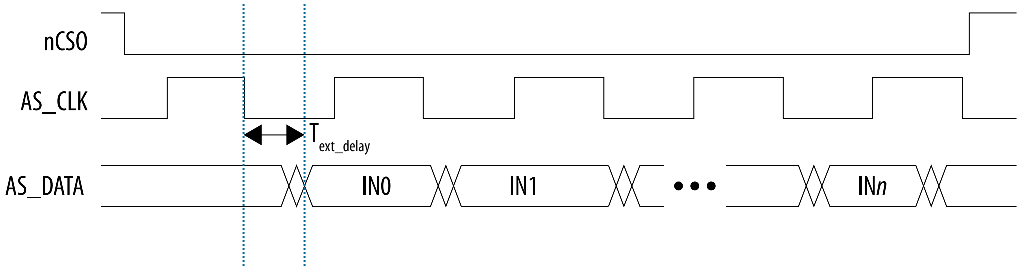


Figure 24. AS Configuration Serial Input Timing Diagram



Note: For more information about the timing parameters, refer to the *Intel Stratix 10 Device Datasheet*.

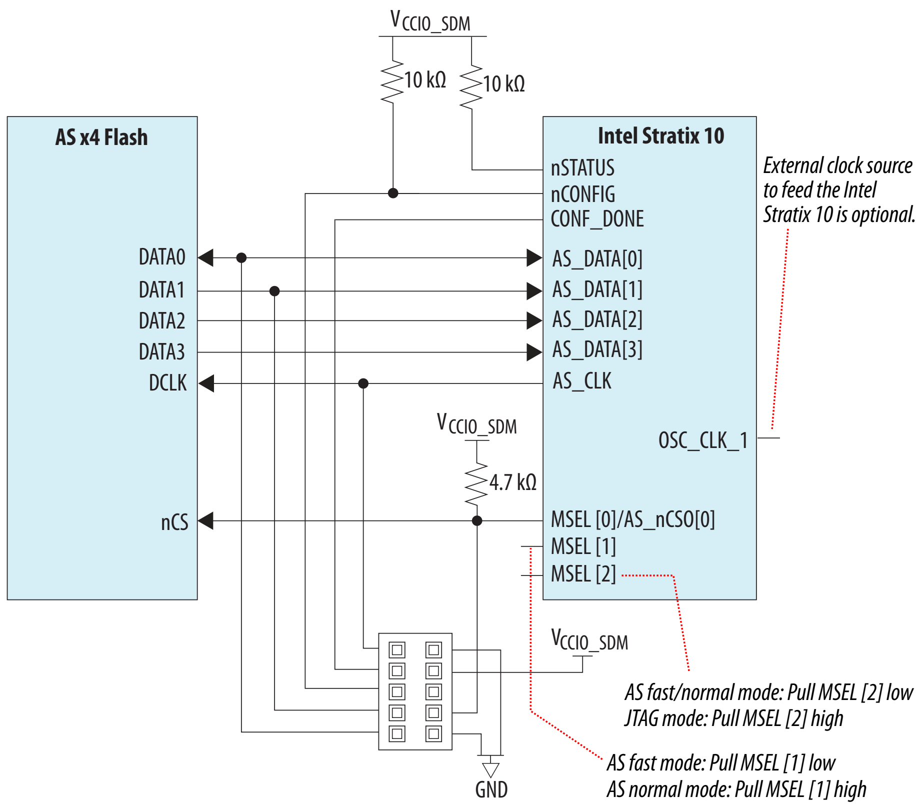
3.2.4. Programming Serial Flash Devices

You can program serial flash devices in-system using the Intel FPGA Download Cable II or Intel FPGA Ethernet Cable.

You have the following two in-system programming options:

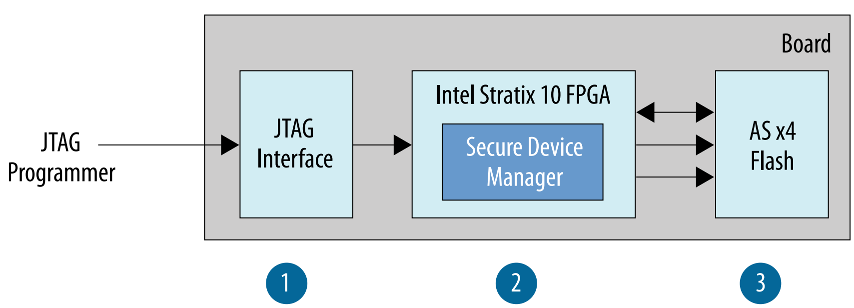
- AS: The Intel Quartus Prime software or any supported third-party software programs the configuration data directly into the serial flash device. You must set MSEL to JTAG. When MSEL is set to JTAG, the SDM tristates the AS pins allowing the Intel Quartus Prime Programmer to program the flash memory devices via the AS header.

Figure 25. AS Programming Using Intel Quartus Prime or Third-Party Programmer



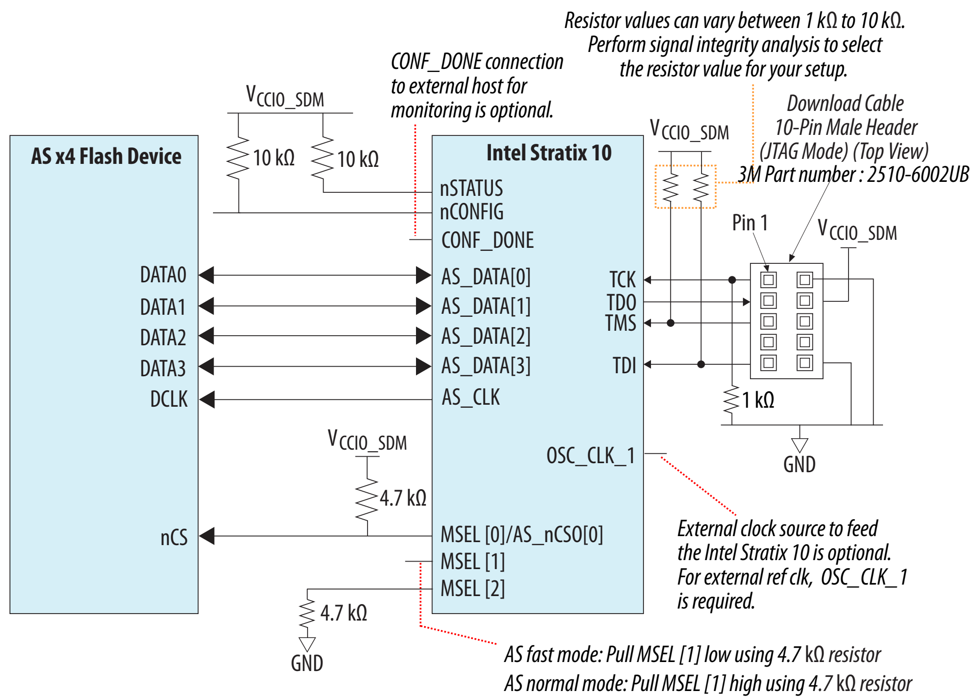
- JTAG: The Intel Quartus Prime Programmer interfaces to the SDM device through JTAG interface and programs the serial flash device.

Figure 26. Programming Your Serial Configuration Device Using JTAG and SDM Emulation of AS



3.2.4.1. Programming Serial Flash Devices using the JTAG Interface

Figure 27. Connection Setup for Programming the Serial Flash Devices using the JTAG Interface



Intel recommends using the JTAG interface to prepare the QSPI flash device for later use in AS mode. Set the MSEL mode to JTAG for when programming the AS x4 device with a `.jic` file.

This configuration scheme includes the following steps:

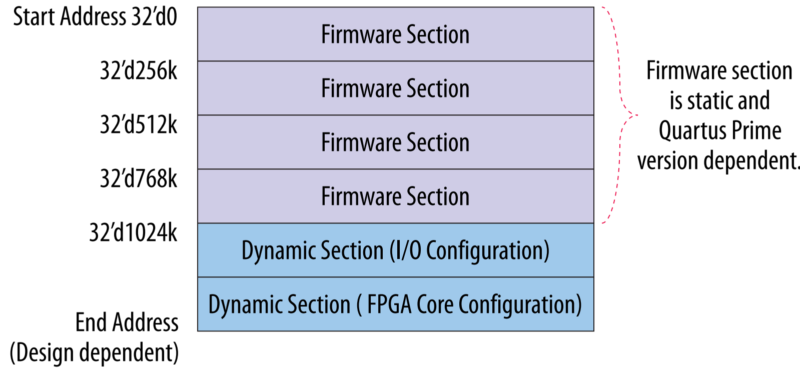
1. In the Intel Quartus Prime Programmer, select the **JTAG** programming mode and initiate programming by clicking **Start**.
2. The Programmer drives `.jic` configuration data to the board using the JTAG header connection.
3. The programmer first configures the SDM with configuration firmware. Then, the SDM drives configuration data from the programmer to the AS x4 flash device using SDM_IOs.
4. To use the Intel Stratix 10 device in AS mode after successful programming of the flash device, set the MSEL `pins to either AS fast or AS normal mode and power cycle the device.`

3.2.5. Serial Flash Memory Layout

Serial flash devices store the configuration data in sections.

The following diagram illustrates sections of a non-HPS Intel Stratix 10 configuration data mapping in serial flash device. Refer to *Intel Stratix 10 SoC FPGA Bitstream Sections* of the *HPS Technical Reference Manual* for more information about flash memory layout for HPS devices.

Figure 28. Serial Flash Memory Layout Diagram



If you use a third-party programmer to program an `.rpd`, ensure that the configuration data is stored starting from address 0 of the serial flash device. If you use `.jic` or `.pof` files, the Intel Stratix 10 Programmer automatically programs the configuration data starting from address 0 of the serial flash device.

Intel currently support the following third-party flash devices:

- Micron MT25Q 512 megabytes (MB)
- Macronix MX66U 512 MB, 1 and 2 gigabytes (GB)
- Macronix MX25U 128 MB, 256 MB, and 512 MB
- Micron MT25QU 128 MB, 256 MB, 512 MB, 1 GB, and 2 GB

Related Information

Intel Stratix 10 SoC FPGA Bitstream Sections

3.2.6. AS_CLK

The Intel Stratix 10 device drives AS_CLK to the serial flash device. An internal oscillator or the external clock that drives the OSC_CLK_1 pin generates AS_CLK. Using an external clock source allows the AS_CLK to run at a higher frequency. If you provide a 25 MHz, 100 MHz, or 125 MHz clock to the OSC_CLK_1 pin, the AS_CLK can run up to 133 MHz. Set the maximum required frequency for the AS_CLK pin in the Intel Quartus Prime software as described in Active Serial Configuration Software Settings on page 58. The AS_CLK pin runs at or below your selected frequency.

Table 24. Supported configuration clock source and AS_CLK Frequencies in Intel Stratix 10 Devices

Configuration Clock Source	AS_CLK Frequency (MHz)
Internal oscillator	• 115 • 77 • 58 • 25
OSC_CLK_1	• 25 • 50 • 80

continued...

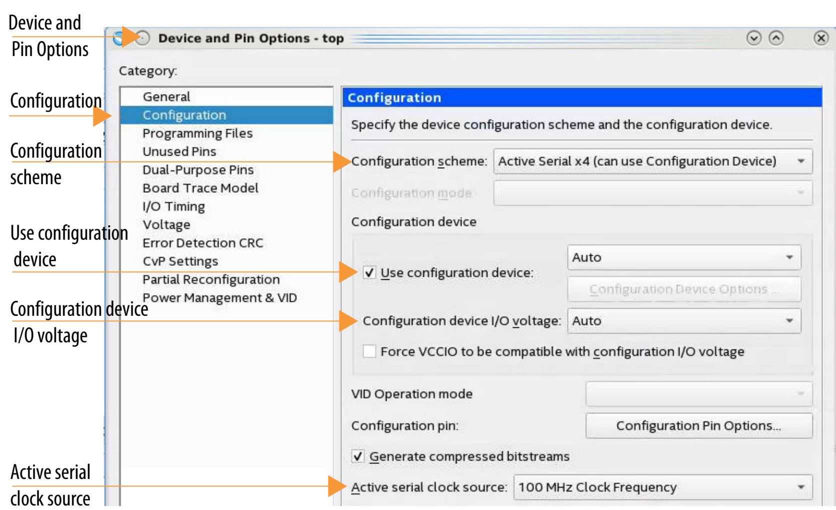
Configuration Clock Source	AS_CLK Frequency (MHz)
	• 100 • 108 • 125 • 133

3.2.7. Active Serial Configuration Software Settings

You must set the parameters in the **Device and Pin Options** of the Intel Quartus Prime software when using the AS configuration scheme.

To set the parameters for AS configuration scheme, complete the following steps:

1. On the **Assignments** menu, click **Device**.
2. In the **Device and Pin Options** select the **Configuration** category.
    a. Select **Active Serial x4** from the **Configuration scheme** drop down menu.



    b. Turn on the **Use configuration device** and select your serial flash device from the drop-down list.
    c. Select **Auto** or **1.8 V** in the **Configuration device I/O voltage** drop-down list.
    d. Select the AS clock frequency from the **Active serial clock source** drop-down list.
3. Click **OK** to confirm and close the **Device and Pin Options**.

Related Information

Can I use 3rd party QSPI flash devices for Active Serial configuration of Intel Stratix 10 devices?

3.2.8. Generating and Programming AS Configuration Programming Files

You must perform the following steps before configuring the Intel Stratix 10 using AS configuration scheme:

1. Generate `.pof`, `.jic`, or `.rpd` programming files using Convert Programming Files
2. Program the `.pof`, `.jic`, or `.rpd` file into the serial flash.

Note:
- You can use the Intel Quartus Prime Programmer to program the `.pof` or `.jic` file into the serial flash device through an AS header or JTAG interface respectively. Alternatively, you can use a third-party programmer to program the `.rpd` file into the serial flash device.
- Refer to the related information for more information about enabling other flash device support.

Related Information

Can I use 3rd party QSPI flash devices for Active Serial configuration of Intel Stratix 10 devices?

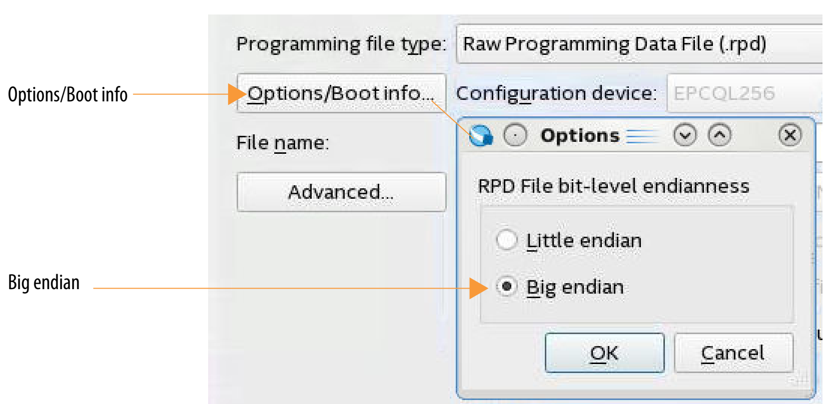
3.2.8.1. Generating Programming Files using Convert Programming Files

The Intel Quartus Prime **Convert Programming File** dialog box converts the `.sof` input file to a `.pof`, `.jic`, or `.rpd` file.

To convert the programming files, complete the following steps:

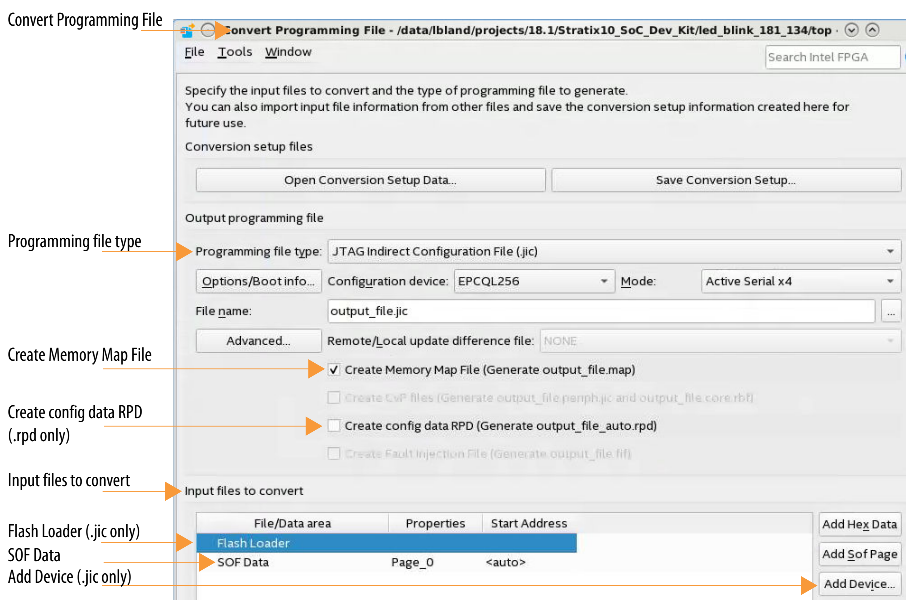
1. On the **File** menu, click **Convert Programming Files**.
2. Under **Output programming file**, select appropriate file type for your design. The AS scheme supports the **Programmer Object File (.pof)**, **JTAG Indirect Configuration File (.jic)**, and **Raw Programming Data File (.rpd)** file types.
3. In the **Mode** list, select **Active Serial x4**.
4. By default, the `.rpd` file type is little-endian, if you are using a third-party programmer that does not support the little-endian format, click **Option/Boot Info** button. In the **Options** dialog box, set the RPD File Endianness to **Big Endian**.

Figure 29. Specifying RPD Bit-Level Endianness



5. In the **File name** field, specify the file name for the programming file you want to create.
6. Under **Advanced** to generate a Memory Map File (`.map`), turn on **Create Memory Map File (Generate output_file.map)** This option is not available for `.rpd` files.
7. To generate a Raw Programming Data (`.rpd`), turn on **Create config data RPD (Generate output_file_auto.rpd)**.
8. For `.jic` output, select **Flash Loader** and click **Add device**. Select your device family and device name and click **OK**.
9. You can add the `.sof` on the **Input files to convert** list.

Figure 30. AS Convert Programming File Options for .jic Generation



10. For `.rpd` generation, you can add the `.pof` file in the **Input files to convert** list as the source file to generate the `.rpd` file.
11. Click **Generate** to generate related programming file.

3.2.8.2. Programming .pof files into Serial Flash Device

To program the `.pof` into the serial flash device through the AS header, perform the following steps:

1. In the **Programmer** window, click **Hardware Setup** and select the desired download cable.
2. In the **Mode** list, select **Active Serial Programming**.
3. Click **Auto Detect** button on the left pane.
4. Select the device to be programmed and click **Add File**.

5. Select the `.pof` to be programmed to the selected device.
6. When available, you can enable the real-time ISP mode by turn-on the **Enable real-time ISP to allow background programming**.
7. Click **Start** to start programming.

3.2.8.3. Programming .jic files into Serial Flash Device

To program the `.jic` into the serial flash device through the JTAG interface, perform the following steps:

1. In the **Programmer** window, click **Hardware Setup** and select the desired download cable.
2. In the **Mode** list, select **JTAG**.
3. Select the device to be programmed and click **Add File**.
4. Select the `.jic` to be programmed to the selected device.
5. Click **Start** to start programming.

3.2.9. Debugging Guidelines for the AS Configuration Scheme

The AS configuration scheme operation is like earlier device families. However, there is one significant difference. Intel Stratix 10 devices using AS mode, try to load a firmware section from addresses 0, 256k, 512k and 768k in the serial flash device connected to the CS0 pin. The firmware section is static for a particular Intel Quartus Prime Pro Edition release.

The firmware includes a pointer to the configuration bitstream design sections. If the configuration bitstream does not include a valid image, the SDM asserts an error by driving nSTATUS low. You can recover from the error by reconfiguring the FPGA over JTAG, or by driving `nCONFIG` low.

SDM tristates AS pins, `AS_CLK`, `AS_DATA0`-`AS_DATA3`, and `AS_CS0`-`AS_CS3`, only when the device powers on if you set `MSEL` to JTAG. If `MSEL` is either AS fast or normal, the SDM drives the AS pins until you power cycle the Intel Stratix 10 device. Unlike earlier device families, the AS pins are not tristated when the device enters user mode.

The AS configuration scheme has power-on requirements. If you use AS Fast mode and are not concerned about 100 ms PCIe link training requirement, you must still ramp the V_{CCIO_SDM} supply within 18 ms. This ramp-up requirement ensures that the AS x4 device is within its operating voltage range when the Intel Stratix 10 device begins assessing the AS x4 device.

When using AS fast mode, all power supplies to the Intel Stratix 10 device must be fully ramped-up to the recommended operating conditions within 10 ms. To meet the PCIe 100 ms power-up-to-active time requirement for CvP, the V_{CCIO_SDM} power to the Intel Stratix 10 device must be at the recommended operating range within 10 ms.

Debugging Suggestions

Here are some debugging tips for the AS configuration scheme:

- Ensure that the boot address for your configuration image is correctly defined when generating the programming file for the flash. The boot address defaults to 0 for AS configuration.
- Ensure that the design meets the power-supply ramp requirements for fast AS mode. If using fast mode, V_{CCIO_SDM} must ramp up within 18 ms.
- Ensure that the flash is powered up and ready to be accessed when the Intel Stratix 10 device exists power-on reset.
- If you are using an external clock source for configuration, ensure the OSC_CLK_1 pin is fed correctly, and the frequency matches the frequency you set for the OSC_CLK_1 in your Intel Quartus Prime Pro Edition project.
- Ensure the MSEL pins reflect the correct AS configuration scheme.
- If the AS configuration is failing due to a corrupt image inside the serial flash device, change the MSEL pins to JTAG only mode, verify that configuration is successful over JTAG. Then, erase and reprogram the serial flash device.
- If you are using AS x4 flash memories, ensure that you use AS Fast mode, if you are not concerned about 100 ms PCIe linkup, you must still ramp the V_{CCIO_SDM} supply within 18 ms. This ramp-up requirement ensures that the AS x4 device is within its operating voltage range when the Intel Stratix 10 device begins to access it.

3.3. Configuration from SD MMC

Note: Contact your Intel sales representative for information about SD MMC support.

In the configuration scheme using SD memory cards, or MMC, the memory cards store configuration. The SDM uses the on-chip SD or MMC controller to interface to the memory cards. The SDM block reads the configuration data from the memory cards for the configuration process. The configuration from SD and MMC supports x4 SD memory cards and x8 MMC.

Table 25. Intel Stratix 10 Configuration Data Width, Clock Rates, and Data Rates

Mode		Data Width (bits)	Max Clock Rate	Max Data Rate	MSEL[2:0]
Active	SD/MMC	4 or 8	50 MHz	400 Mbps	100

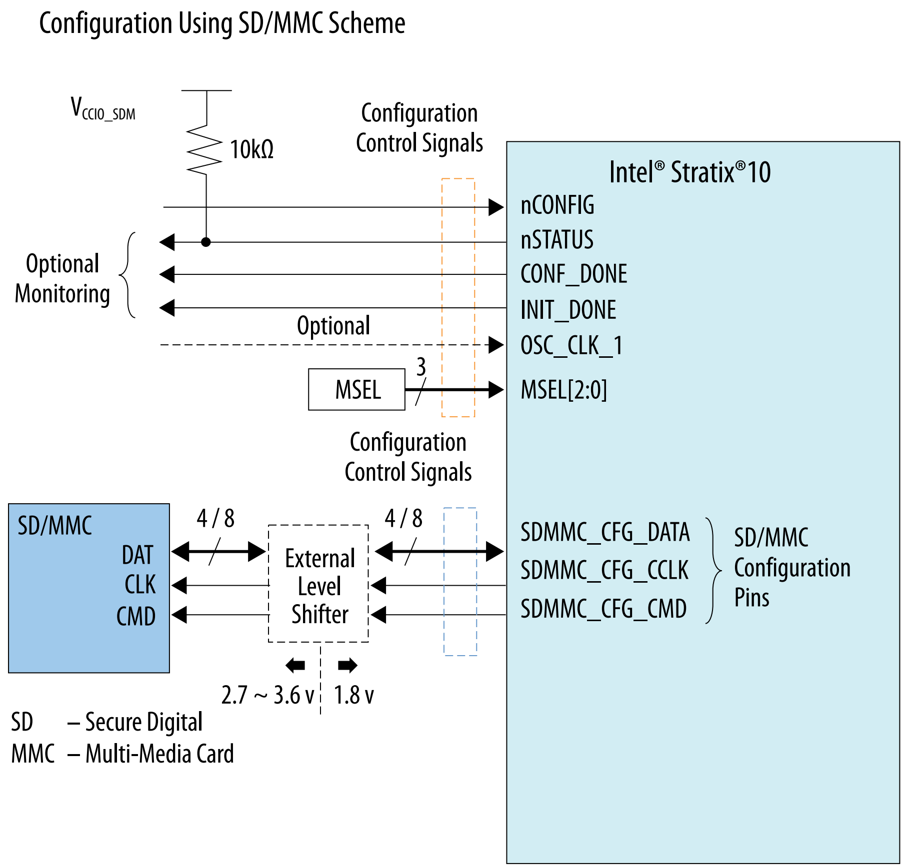
Related Information

- MSEL Settings on page 18
- SD MCC Configuration Timing in Intel Stratix 10 Devices

3.3.1. SD MMC Single-Device Configuration

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* for additional information about individual pin usage and requirements.

Figure 31. Connections for SD MMC Single-Device Configuration



Note: The External Level Shifter is not mandatory for embedded multimedia cards (eMMC).

Related Information

Intel Stratix 10 Device Family Pin Connection Guidelines

3.4. JTAG Configuration

JTAG-chain device programming is ideal during development. You can reconfigure Intel Stratix 10 using JTAG faster than you can reprogram flash memory. You can also use JTAG to reprogram a corrupted flash memory that is preventing the Intel Stratix 10 device from configuring using its normal configuration scheme. The Intel Quartus Prime software generates a `.sof` for JTAG configuration. Use the Intel Quartus Prime Programmer with the Intel FPGA download cable to configure the Intel Stratix 10 device through its JTAG interface. The Intel FPGA Download Cable II and the Intel FPGA Ethernet Cable can support the V_{CCIO_SDM} supply at 1.8 V. Alternatively, you can use the Jam*STAPL Format File (`.jam`) or Jam Byte Code File (`.jbc`) with other third-party programmer tools.

Intel Stratix 10 devices automatically compress the configuration bitstream. You cannot disable compression in Intel Stratix 10 devices.

Table 26. Intel Stratix 10 Configuration Data Width, Clock Rates, and Data Rates

Mode		Data Width (bits)	Max Clock Rate	Max Data Rate	MSEL[2:0]
Passive	JTAG	1	30 MHz	30 Mbps	3'b111

Related Information

- Programming Support for Jam STAPL Language
- JTAG Configuration Timing in Intel Stratix 10 Devices

3.4.1. JTAG Single-Device Configuration

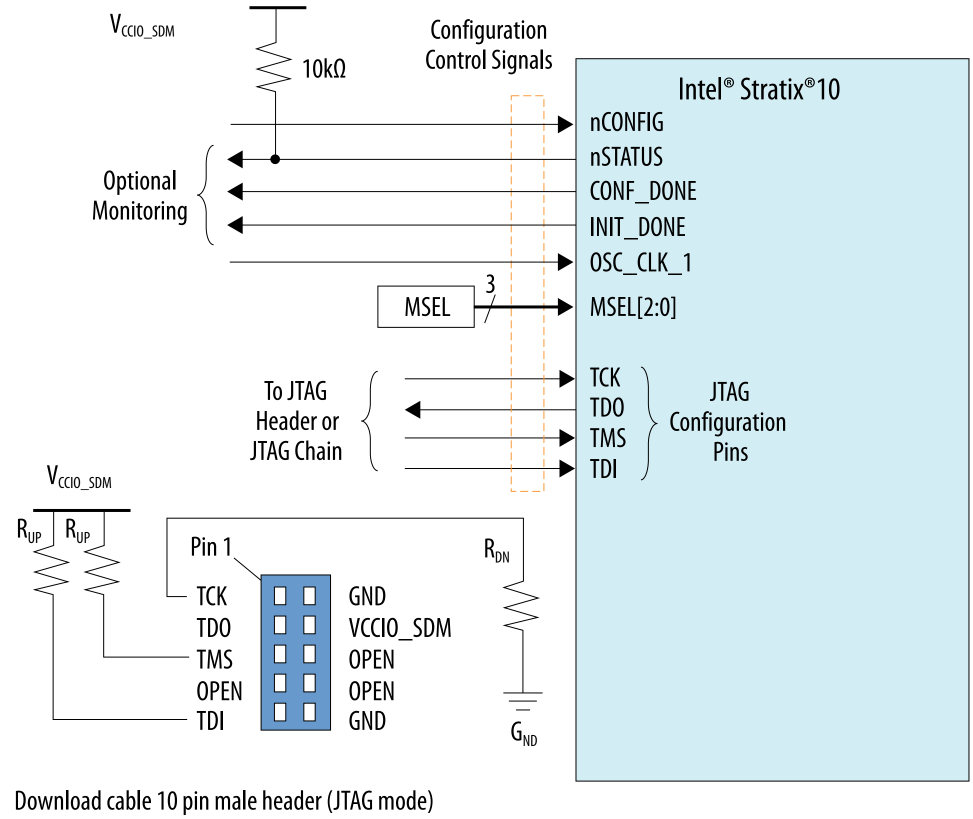
To configure a single device in a JTAG chain, the programming software sets the other devices to bypass mode. A device in bypass mode transfers the programming data from the TDI pin to the TDO pin through a single bypass register. The configuration data is available on the TDO pin one clock cycle later.

You can configure the Intel Stratix 10 device through JTAG using a download cable or a microprocessor.

3.4.1.1. JTAG Single-Device Configuration using Download Cable Connections

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* for additional information about individual pin usage and requirements.

Figure 32. Connection Setup for JTAG Single-Device Configuration using Download Cable



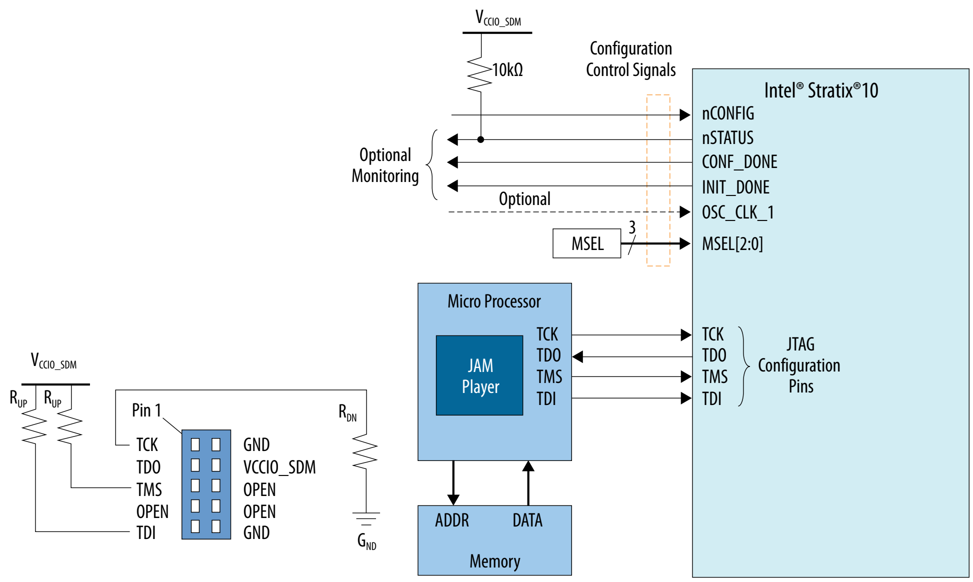
Related Information

- Intel FPGA Download Cable II User Guide
- Intel Stratix 10 Device Family Pin Connection Guidelines

3.4.1.2. JTAG Single-Device Configuration using a Microprocessor

Refer to the *Intel Stratix 10 Device Family Pin Connection Guidelines* for additional information about individual pin usage and requirements.

Figure 33. Connection Setup for JTAG Single-Device Configuration using a Microprocessor



Related Information

Intel Stratix 10 Device Family Pin Connection Guidelines

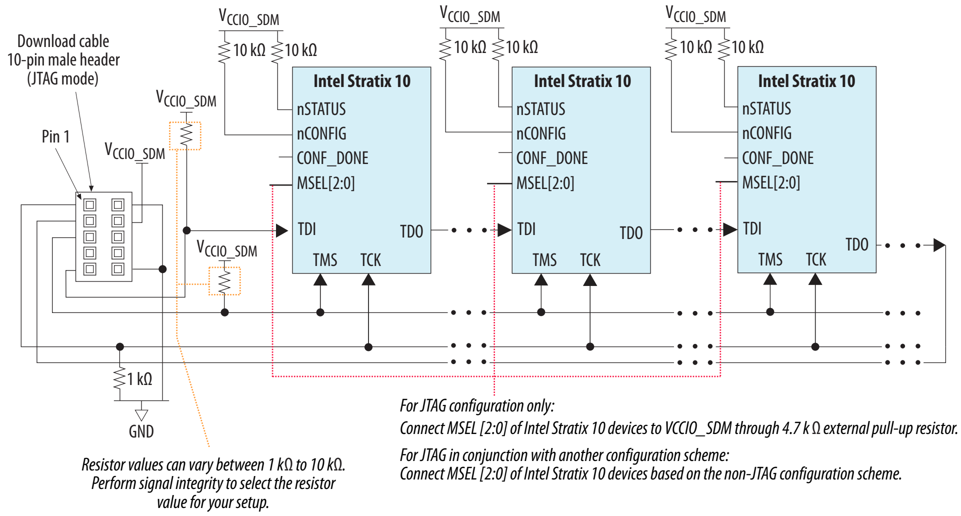
3.4.2. JTAG Multi-Device Configuration

You can configure multiple devices in a JTAG chain. Observe the following pin connections and guidelines for this configuration setup:

- One JTAG-compatible header connects to several devices in a JTAG chain. The drive capability of the download cable is the only limit on the number of devices in the JTAG chain.
- If you have four or more devices in a JTAG chain, buffer the TCK, TDI, and TMS pins with an on-board buffer. You can also connect other Intel FPGA devices with JTAG support to the chain.

3.4.2.1. JTAG Multi-Device Configuration using Download Cable

Figure 34. Connection Setup for JTAG Multi Device Configuration using Download Cable



3.4.3. Debugging Guidelines for the JTAG Configuration Scheme

The JTAG configuration scheme overrides all other configuration schemes. The SDM is always ready to accept configuration over JTAG unless a security feature disables the JTAG interface. JTAG is particularly useful in recovering a device that may be in an unrecoverable state reached when trying to configure using a corrupted image.

An nSTATUS falling edge terminates any JTAG access and the device reverts to the `MSEL`-specified boot source. nSTATUS must be stable during JTAG configuration. nSTATUS follows `nCONFIG` during JTAG configuration. Consequently, `nCONFIG` also must be stable.

Unlike other configuration schemes, nSTATUS does not assert if an error occurs during JTAG configuration. You must monitor the error messages that the Intel Quartus Prime Pro Edition Programmer generates for error reporting.

Debugging Suggestions

Here are some debugging tips for JTAG:

- If JTAG configuration is failing, check that the FPGA has successfully powered up and exited POR. One way is to check the hand shaking behavior between `nCONFIG` and nSTATUS by driving `nCONFIG` low and ensuring that nSTATUS also goes low.
- Another way to determine whether the device has exited the POR state is to use the Intel Quartus Prime Programmer to detect the device. If the programmer can detect the Intel Stratix 10 device, it has exited the POR state.
- If using an Intel FPGA Download Cable II, reduce the cable clock speed to 6 MHz.

- If you have multiple devices in the JTAG chain, try to disconnect other devices from the JTAG chain to isolate the Intel Stratix 10 device.
- If you specify the OSC_CLK_1 as the clock source for configuration, ensure that OSC_CLK_1 is running at the frequency you specify in the Intel Quartus Prime software.
- For designs including the High Bandwidth Memory (HBM2) IP or any IP using transceivers, you must provide a free running and stable reference clock to the device before device configuration begins. All transceiver power supplies must be at the required voltage before configuration begins.

4. Stratix 10 Configuration Features

4.1. Device Security

Note: Contact your Intel sales representative for more information about the device security support in Intel Stratix 10 devices.

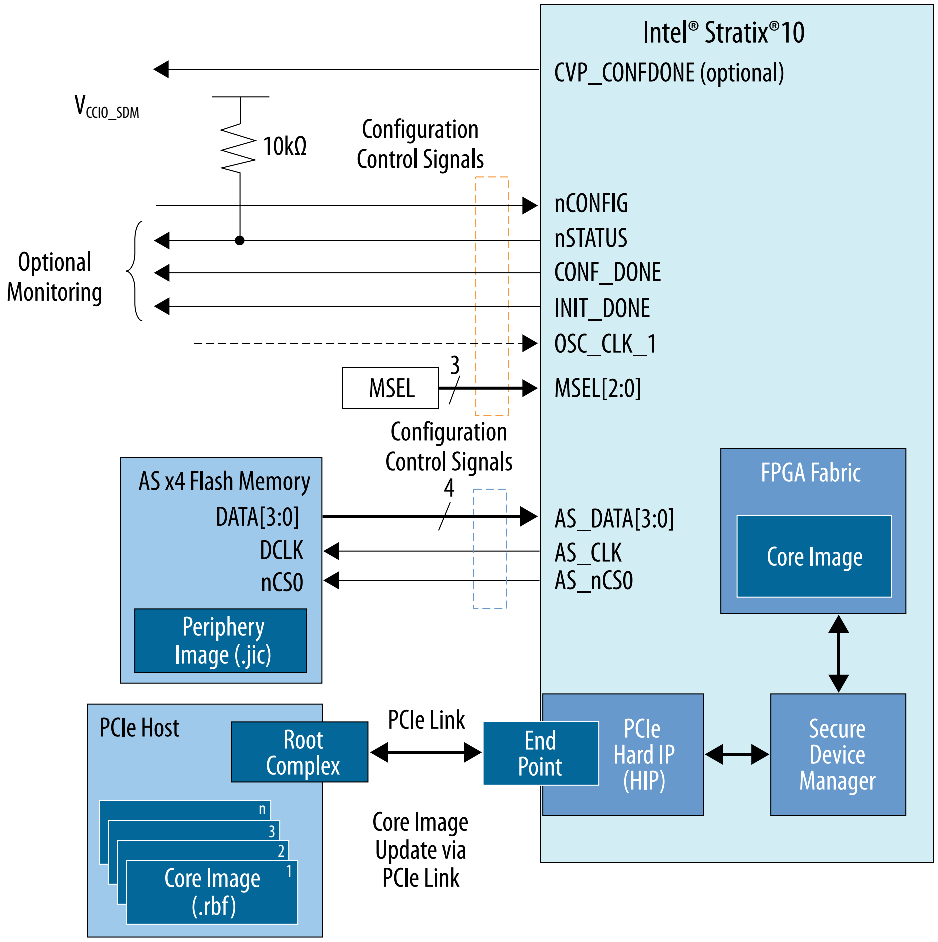
The Intel Stratix 10 device provides the following flexible and robust security features to protect sensitive data and intellectual property:

- User image authentication and encryption
- Public-Key based authentication
- Advanced Encryption Standard (AES)-256 Encryption
- JTAG Disable
- JTAG Debug Disable/Enable
- Side channel protection
- Physical anti-tampering

4.2. Configuration via Protocol

The CvP configuration scheme creates separate images for the periphery and core logic. You can store the periphery image in a local configuration device and the core image in host memory, reducing system costs and increasing the security for the proprietary core image. CvP configures the FPGA fabric through the PCI Express* (PCIe) link and is available for Endpoint variants only.

Figure 35. Intel Stratix 10 CvP Configuration Block Diagram



The CvP configuration scheme supports the following modes:

- CvP Initialization Mode:

  In this mode an external configuration device stores the periphery image and it loads into the FPGA through the Active Serial x4 (Fast mode) configuration scheme. The host memory stores the core image and it loads into the FPGA through the PCIe link.

  After the periphery image configuration completes, the CONF_DONE signal goes high and the FPGA starts PCIe link training. When PCIe link training completes, the PCIe link transitions to the Link Training and Status State Machine (LTSSM) L0 state and then through PCIe enumeration. The PCIe host then configures the core through the PCIe link. The PCIe reference clock must be running for the link for link training.

  After the core image configuration is complete, the CvP_CONFDONE pin (if enabled) goes high, indicating the FPGA is fully configured.

- CvP Update Mode

  CvP update mode is a reconfiguration scheme that allows an FPGA device to deliver an updated bitstream to a target device after the device enters user mode. In this mode, the FPGA device initializes by loading the full configuration image from the external local configuration device to the FPGA or after CvP initialization.

  You can perform CvP update on a device that you originally configure using CvP initialization or any other configuration scheme.

Related Information

Intel Stratix 10 Configuration via Protocol (CvP) Implementation User Guide

4.3. Partial Reconfiguration

Partial reconfiguration (PR) allows you to reconfigure a portion of the FPGA dynamically, while the remaining FPGA design continues to function. You can define multiple personas for a region in your design, without impacting operation in areas outside this region. This methodology is effective in systems with multiple functions that time-share the same FPGA device resources. PR enables the implementation of more complex FPGA systems.

Related Information

Intel Quartus Prime Pro Edition User Guide: Partial Reconfiguration

5. Remote System Upgrade

Remote system upgrade implements device reconfiguration using dedicated remote system upgrade circuitry available in all Intel Stratix 10 devices. Remote system upgrade has the following advantages:

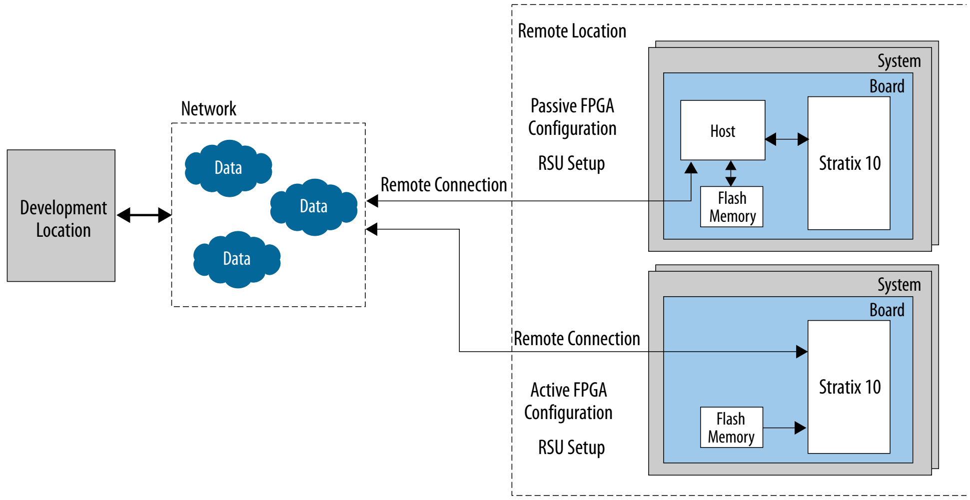
- Provides a mechanism to deliver feature enhancements and bug fixes without recalling your products
- Reduces time-to-market
- Extends product life

Using remote system upgrade, you use the Intel Stratix 10 Serial Flash Mailbox Client Intel FPGA IP to write configuration bitstreams to the AS x4 flash device. Then you can use the Mailbox Client Intel Stratix 10 FPGA IP to instruct the SDM to reboot from the updated image. You can store multiple application images and a single factory image in the configuration device. Your design manages remote upgrades of the application images in the configuration device.

A command to the Mailbox Client Intel Stratix 10 FPGA Mailbox Client IP Core initiates reconfiguration. The remote upgrade system performs configuration error detection during and after the reconfiguration process. If errors in the application images prevent reconfiguration, the configuration circuitry reverts to the default factory image and provides error status information.

The following figure shows functional diagrams for typical remote system upgrade processes. For passive configuration schemes, the host implements remote system update rather than the Intel Stratix 10 device. To learn more about remote system update for passive configuration schemes, refer to *Altera Remote Update IP Core User Guide* for remote system update implementations in earlier device families. This document explains the remote system update implementation for active configuration schemes.

Figure 36. Typical Remote System Upgrade Process



Related Information

Altera Remote Update IP Core User Guide

5.1. Remote System Upgrade Functional Description

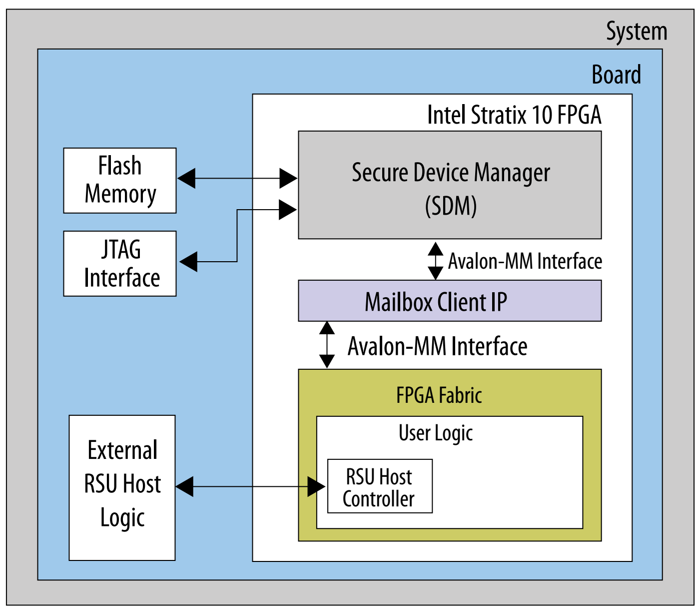
5.1.1. Remote System Upgrade Using AS Configuration

Remote system upgrade using AS configuration includes the following components:

- Your external remote system upgrade host design. The host can be custom logic, the HPS, or a Nios® II processor in the FPGA.
- One factory image.
- Flash memory for image storage.
- At least one application image.

  Note: Remote system upgrade cannot use partial reconfiguration (PR) images for the application image.
- Designs that do not use the HPS as the remote system upgrade host require an Intel Stratix 10 Serial Flash Mailbox Client FPGA IP core as shown in the figure below. The Serial Mailbox Client sends and receives remote system upgrade operation commands and responses.

Figure 37. Intel Stratix 10 Remote System Upgrade Components



Related Information

Mailbox Client Intel Stratix 10 FPGA IP Core User Guide User Guide

5.1.2. Remote System Upgrade Configuration Images

Intel Stratix 10 devices using remote system upgrade require the following configuration images:

- A Factory image—contains logic with enough functionality to implement the following functions:
  - Your design-specific logic to obtain new application images
  - Your design-specific logic to request reconfiguration using a specific application image
  - Image storage in flash memory
- Application image—contains logic to implement the custom application. The application image must also contain logic to obtain new application images and store the images in the flash memory.

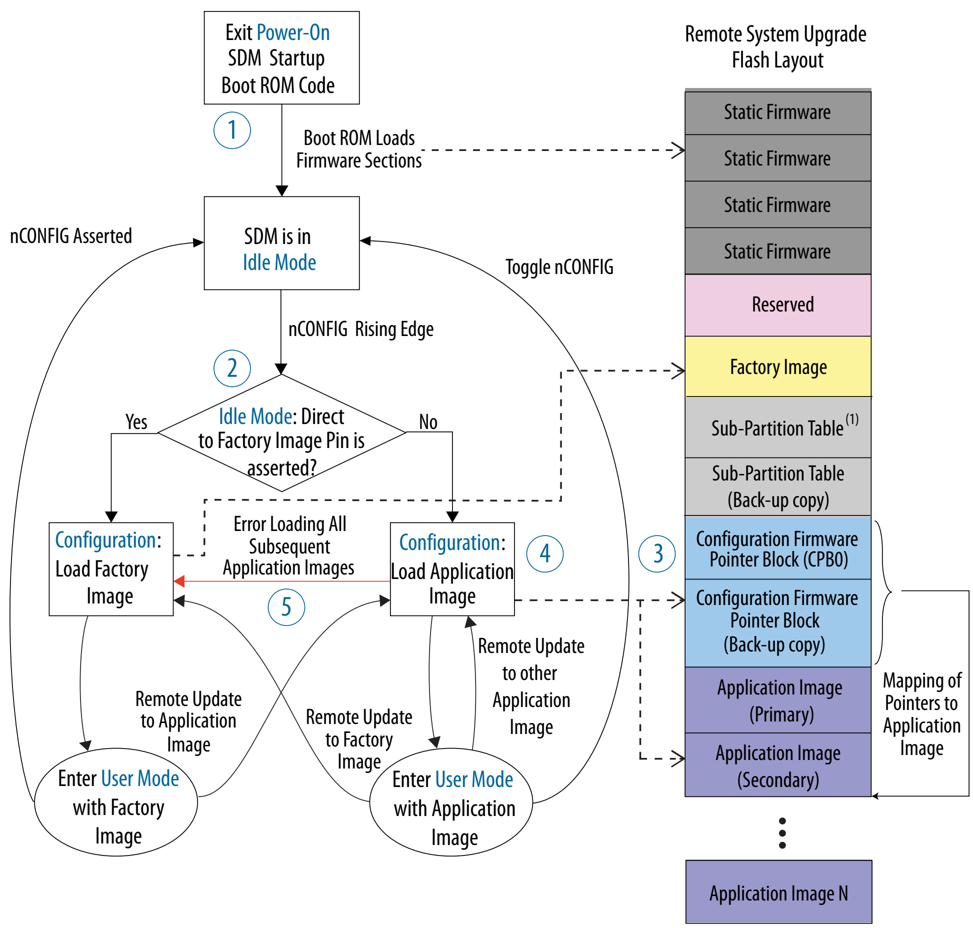
  Note: The application image is optional. You can create a remote system upgrade image containing the factory image only. The application image can be added or obtained after you configure the device with the factory image.

Depending on the storage space of your flash memory, Intel Stratix 10 remote system upgrade supports one factory application image and up to 507 application images. The Quartus Programming File Generator only supports up to three remote system upgrade images. However, you can add new additional images using the Serial Flash Mailbox Client IP with the device in user mode.

5.1.3. Remote System Upgrade Configuration Sequence

Figure 38. Remote System Upgrade Configuration Sequence

In the following figure the blue text are states shown in the Configuration Flow Diagram on page 12.



Reconfiguration includes the following steps:

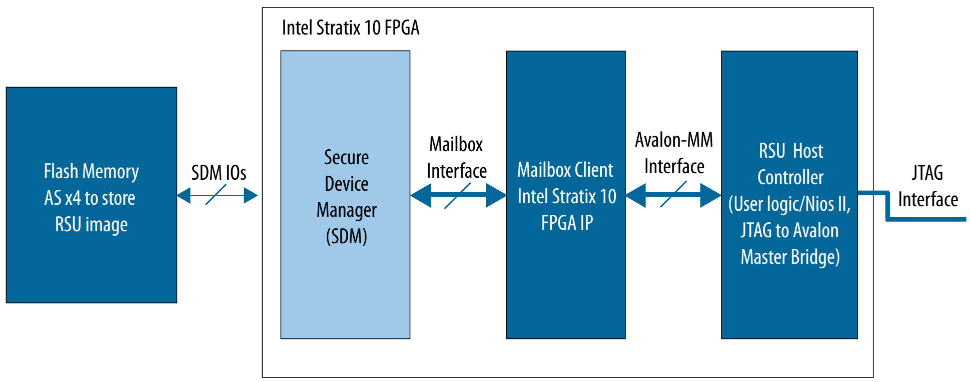
1. After the device exits power-on-reset (POR), the boot ROM loads flash memory from one of the four static firmware slots at addresses 0, 256k, 512k, or 768k to initialize the SDM. The same configuration firmware is present in each of these locations. (Refer to Step 2 of Guidelines for Performing Remote System Upgrade Functions for Non-HPS on page 77 for step-by-step details for programming the firmware into the flash.)
2. The optional direct-to-factory pin controls whether the SDM firmware loads the factory or application image. You can assign the direct-to-factory input to any unused SDM pin. The SDM loads the application image if you do not assign this pin.
3. The configuration firmware pointer block in the flash device maintains a list of pointers to the application images.

4. When loading an application image, the SDM traverses the pointer block in reverse order. The SDM loads the highest priority image. When image loading completes, the device enters user mode.
5. If loading the newest (highest priority) image is unsuccessful, the SDM tries the next application image from the list. If none of the application loads successfully, the SDM loads the factory image.
6. If loading the factory image fails, you can recover by reprogramming the QSPI flash with the RSU image using the JTAG interface.

If reconfiguring the device with an application image in user mode is unsuccessful, the SDM loads the last working image.

5.2. Guidelines for Performing Remote System Upgrade Functions for Non-HPS

Figure 39. Intel Stratix 10 Modules and Interfaces to Implement RSU Using Images Stored in Flash Memory



Note: Refer to the *Intel Stratix 10 SoC Development Kit User Guide* for more details on using HPS as the RSU host to perform remote system upgrade.

Here are guidelines to follow when implementing remote system upgrade:

1. The factory or application image must at least contain a remote system upgrade host controller and a Mailbox Client Intel Stratix 10 FPGA IP.
    - You can use either custom logic, the Nios II processor, or the JTAG to Avalon Master Bridge IP as a remote system upgrade host controller.
    - The remote system upgrade host controller controls the remote system upgrade function by sending commands to and receiving responses from the SDM via Mailbox Client Intel Stratix 10 FPGA IP. The Mailbox Client functions as the messenger between the remote system upgrade host and SDM. It passes the commands to and responses from the SDM.
2. The pre-generated standard remote system upgrade image file should include either a factory image or a factory image and at least one application image. The remote system upgrade image must be programmed into the flash memory. In user mode you can program additional application images.

- Refer to Generating Remote System Upgrade Image Files using Programming File Generator on page 84 for the step by step process to generate the standard and single remote system upgrade image files using the programming file generator.

3. The remote system update requires you to use the AS x4 configuration scheme to configure the FPGA with the pre-generated remote system upgrade image.
4. Once the device enters user mode with either the factory image or an application image, the remote system upgrade host can perform the following remote system upgrade operations:
   a. Reconfiguring the device with an application or factory image
      i. From factory image to an application image or vice versa
      ii. From an application image to another application image
   b. Erasing the application image
   c. Adding an application image

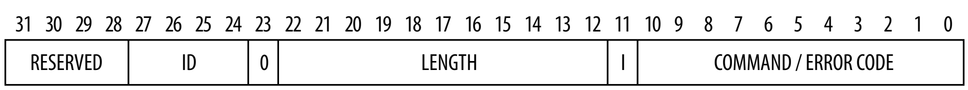
Related Information

- Intel Stratix 10 SoC Development Kit User Guide
- Mailbox Client Intel Stratix 10 FPGA IP Core User Guide

5.3. Commands and Error Codes

The remote system upgrade host communicates with the SDM using command and response packets via the Mailbox Client Intel Stratix 10 FPGA IP.

Figure 40. Command and Error Code Header Format



The following table describes the fields of the header command.

Table 27. Mailbox Client Intel Stratix 10 FPGA IP Command and Error Code Header Description

Header	Bit	Description
Reserved	[31:28]	Reserved.
`ID`	[27:24]	The command ID. The response header returns the ID specified in the command header. Set different IDs in each command to match responses with commands.
`0`	[23]	Reserved.
`Length`	[22:12]	Number of words of arguments following the header.
`I`	[11]	Mailbox Client. Set this bit to `0` when sending commands using the Mailbox Client Intel Stratix 10 FPGA IP core.
`Command Code/Error Code`	[10:0]	`Command Code` specifies the command. The `Error Code` indicates whether the command succeeded or failed.

5.3.1. Operation Commands

Table 28. Mailbox Client Intel Stratix 10FPGA IP Command List and Description

<table>
<tr><th>Command</th><th>Code (Hex)</th><th>Number of Commands [13]</th><th>Number of Responses [13]</th><th>Description</th></tr>
<tr><td>RSU_IMAGE_UPDATE</td><td>5C</td><td>2</td><td>0</td><td>Triggers reconfiguration from data source or configuration data stored in AS x4 flash memory.
This command takes a 64-bit argument to specify the reconfiguration address in flash memory.
• Bit [63:32]: Reserved (write as 0).
• Bit [31:0]: The start address of an application image.
Returns non-zero response if the device is already processing a configuration.</td></tr>
<tr><td>CONFIG_STATUS</td><td>4</td><td>0</td><td>6</td><td>Reports the status of the last reconfiguration. You can use this command to check the configuration status during and after configuration. The response contains the following fields:
<table>
<tr><th>Word</th><th>Summary</th><th>Description</th></tr>
<tr><td>0</td><td>State</td><td>Describes the most recent configuration related error. 0 when no configuration errors.
The error field has 2 fields:
• Upper 16 bits: Major error code
• Lower 16 bits: Minor errors that do not contain meaningful data.
Here are valid values for major error codes:
<table>
<tr><th>Major Error Code</th><th>Description</th></tr>
<tr><td>0xF001</td><td>BITSTREAM_ERROR</td></tr>
<tr><td>0xF002</td><td>HARDWARE_ACCESS_FAILURE</td></tr>
<tr><td>0xF003</td><td>BITSTREAM_CORRUPTION</td></tr>
<tr><td>0xF004</td><td>INTERNAL_ERROR</td></tr>
<tr><td>0xF005</td><td>DEVICE_ERROR</td></tr>
<tr><td>0xF006</td><td>HPS_WATCHDOG_TIMEOUT</td></tr>
<tr><td>0xF007</td><td>INTERNAL_UNKNOWN_ERROR</td></tr>
</table></td></tr>
<tr><td>1</td><td>Version</td><td>0 for this version.</td></tr>
<tr><td>2</td><td>Pin status</td><td>Specifies the values of the following configuration control signals.
• Bit [31]: Current nSTATUS output value (active low).
• Bit [30]: Detected nCONFIG input value (active low).
• Bit [29:8]: Reserved.
• Bit [7:0]: The MSEL value at power up.</td></tr>
<tr><td>3</td><td>Soft function status</td><td>Specifies the value of each of the soft functions, whether you assigned the function assigned to an SDM pin.</td></tr>
</table></td></tr>
</table>

continued...

(13) The number does not include the command and response header.

<table>
<tr><th>Command</th><th>Code (Hex)</th><th>Number of Commands (13)</th><th>Number of Responses (13)</th><th colspan="4">Description</th></tr>
<tr><td rowspan="3"></td><td rowspan="3"></td><td rowspan="3"></td><td rowspan="3"></td><td></td><td></td><td colspan="2">Bit [31:4]: Reserved.
Bit [3]: SEU_ERROR.
Bit [2]: CVP_DONE.
Bit [1]: INIT_DONE.
Bit [0]: CONF_DONE.</td></tr>
<tr><td>4</td><td>Error location</td><td colspan="2">Contains the error location. Returns 0 for no error.</td></tr>
<tr><td>5</td><td>Error details</td><td colspan="2">Contains the error details. Returns 0 for no error.</td></tr>
<tr><td rowspan="16">RSU_STATUS</td><td rowspan="16">5B</td><td rowspan="16">0</td><td rowspan="16">8</td><td colspan="4">Reports the current remote system upgrade status. This command returns the following responses:</td></tr>
<tr><td>Word</td><td>Summary</td><td colspan="2">Description</td></tr>
<tr><td>0-1</td><td>Current image</td><td colspan="2">Flash offset of the currently running application image.</td></tr>
<tr><td>2-3</td><td>Last failing image</td><td colspan="2">Flash offset of the last failing application image. A value of all 1s indicates no failing images. When there are no failing images, the following words do not contain meaningful data.</td></tr>
<tr><td rowspan="9">4</td><td rowspan="9">State</td><td colspan="2">Failure code of the last failing image.
The error field has two parts:
Upper 16 bits: Major error code.
Lower 16 bits: Minor errors that does not contain meaningful data.
The following major error codes are defined:</td></tr>
<tr><td>Major Error Code</td><td>Description</td></tr>
<tr><td>0xF001</td><td>BITSTREAM_ERROR</td></tr>
<tr><td>0xF002</td><td>HARDWARE_ACCESS_FAILURE</td></tr>
<tr><td>0xF003</td><td>BITSTREAM_CORRUPTION</td></tr>
<tr><td>0xF004</td><td>INTERNAL_ERROR</td></tr>
<tr><td>0xF005</td><td>DEVICE_ERROR</td></tr>
<tr><td>0xF006</td><td>HPS_WATCHDOG_TIMEOUT</td></tr>
<tr><td>0xF007</td><td>INTERNAL_UNKNOWN_ERROR</td></tr>
<tr><td>5</td><td>Version</td><td colspan="2">Contains the value of each of the soft functions, whether that function is on an SDM pin.</td></tr>
<tr><td>6</td><td>Error location</td><td colspan="2">Contains the error location of the last failing image. Returns 0 for no error.</td></tr>
<tr><td>7</td><td>Error details</td><td colspan="2">Contains the error details of the last failing image. Returns 0 for no error.</td></tr>
<tr><td>QSPI_OPEN</td><td>32</td><td>0</td><td>0</td><td colspan="4">Clients use this command to request exclusive access AS x4 interface. The SDM returns the appropriate response:</td></tr>
</table>

continued...

(13) The number does not include the command and response header.

Command	Code (Hex)	Number of Commands (13)	Number of Responses (13)	Description
				• If AS x4 is available, the SDM grants the request by responding OK. • If the SDM is in the process of configuring the device or AS x4 is in use, the SDM returns the error response. *Note:* The SDM grants exclusive access only to the client using this mailbox. Other clients are not able to access AS x4 until it is closed by this client.
QSPI_CLOSE	33	0	0	Closes exclusive access to the AS x4.
QSPI_SET_CS	34	1	0	Selects the flash memory using chip select lines. • Bit [31:28]: Chip selects for flash memories 0-4 using one-hot encoding. • Bit [27:0]: Reserved (write as 0).
QSPI_READ	3A	2	N	Reads the flash memory. Defines the following 2 parameters: • The flash address offset to start a read. (one word). • Number of words to read. When successful returns the OK response, code followed by the data read from the flash memory. When unsuccessful, returns 1 of the following responses: • Returns an error code. • Returns OK when part of the data read from flash memory is incorrect. *Note:* The maximum transfer size is 4 KB. The QSPI_READ command cannot run during configuration.
QSPI_WRITE	39	2+N	0	Writes data to the flash memory. Defines the following 3 parameters: • The word start address in flash memory. • The size in words. • The data. A successful write returns an OK response code. The client may need to issue QSPI_ERASE command before issuing this command to prepare the memory for writing. *Note:* The maximum transfer size is limited to 4 KB. The QSPI_WRITE command cannot run during configuration.
QSPI_ERASE	38	2	0	Erases a sector of the flash memory. Defines the following 2 parameters: • The word start address in flash memory to begin the erasure. The address must be the start address of a sector in the flash memory. • The number of bytes to erase. The erasure size must be a multiple of 64K bytes. A successful erase returns an OK response code.
QSPI_READ_DEVICE_REG	35	2	N	Reads registers from the flash memory. If the data is not a multiple of 4 bytes, it is padded with 0 bytes until the next word boundary. Takes 2 parameters: • The opcode for the read command. • The number of bytes to read. The maximum size is 8 bytes. A successful read returns an OK response code followed by the data read from the device.
QSPI_WRITE_DEVICE_REG	36	2+N	0	Writes to registers on the flash. Takes 3 arguments: • The opcode for the write command. • The number of bytes to write. The maximum size is 8 bytes. • The data. The maximum write is 2 words, padded with 0 to the word boundary.

continued...

(13) The number does not include the command and response header.

Command	Code (Hex)	Number of Commands (13)	Number of Responses (13)	Description
				A successful write returns an OK response code.
QSPI_SEND_DEVICE_OP	37	1	0	Sends a command opcode to flash memory. Takes 1 argument: • The opcode to send the attached flash memory. A successful command returns an OK response code.

5.3.2. Error Code Responses

Table 29. Mailbox Client Intel Stratix 10 FPGA IP Error Code Responses and Description

Value (Hex)	Error Code Response	Description
0	OK	Indicates that the command completed successfully. . Depending on the command delivered to the Mailbox Client, the response error code may not be sufficient to ensure that the operation completed successfully.
1	INVALID_COMMAND	Indicates that the command is in an incorrect format.
2	UNKNOWN_BR	Indicates that the command code is not understood. This error may occur if you have deselected the **Use the factory default helper image** on the **Programmer Tools -> Options** menu.
3	UNKNOWN	Indicates that the command code is not understood by the currently loaded firmware.
100	NOT_CONFIGURED	Indicates that the device is not configured.
1FF	ALT_SDM_MBOX_RESP_DEVICE_BUSY	Indicates that the device is busy.
2FF	ALT_SDM_MBOX_RESP_NO_VALID_RESP_AVAILABLE	Indicates that there is no valid response available.
3FF	ALT_SDM_MBOX_RESP_ERROR	General Error

(13) The number does not include the command and response header.

5.4. Remote System Upgrade Flash Device Layout

The Intel Quartus Prime Programming Files Generator populates the flash memory when you generate the remote system upgrade programming files.

Table 30. Remote System Upgrade Flash Memory Layout

The start of flash address 0, or the A2 partition within a partitioned flash address 0, must be set up as shown in the following table.

Offset	Size (Byte)	Usage	Sub-Partition Name	Sub-Partition Flag	
				Reserved Address (Bit 0)	Read-Only (Bit 1)
0k	256k	Static Firmware Section	BOOT_INFO (remote system upgrade boot image)	YES	YES
256k	256k	Static Firmware Section			
512k	256k	Static Firmware Section			
768k	256k	Static Firmware Section			
1M	64k	Reserved			
1M+64k	Varies	Factory Image	FACTORY_IMAGE	YES	YES
Next	32k	Sub-partition table	SPT0	YES	NO
Next + 32k	32k	Sub-partition table (Back-up copy)	SPT1	YES	NO
Next + 32k	32k	Configuration firmware pointer block	CPB0	YES	NO
Next + 32k	32k	Configuration firmware pointer block (Back-up copy)	CPB1	YES	NO
Varies	Varies	Application image 1	APP_IMAGE1 (14)	NO	NO
Varies	Varies	Application image 2	APP_IMAGE2 (14)	NO	NO
Varies	Varies	Application image N	APP_IMAGEN (14)	NO	NO

5.4.1. Configuration Firmware Pointer Block (CPB)

The configuration firmware accesses the configuration firmware pointer block when performing remote system upgrade. The Intel Quartus Prime Programming Files Generator sets up the initial configuration firmware pointer block. Each copy of the configuration firmware pointer block (CPB0/CPB1) must be exactly 4 KB.

(14) User-assigned sub-partition name.

Table 31. Configuration Firmware Pointer Block Format

The configuration firmware does not load an image if a pointer contains a value of all zeros or all ones.

Offset	Size (Bytes)	Sub-Partition Name	Example Content
0	32	Reserved	
0x20	8	First (lowest priority) image pointer slot (15)	• Bit [31:0]: Applications Image N Start address • Bit [63:32]: Reserved
0x28	8	Second (2nd lowest priority) image pointer slot	• Bit [31:0]: Applications Image 2 Start address • Bit [63:32]: Reserved
And so on	8	—	—
0xFF0	8	Last (highest priority) image pointer	• Bit [31:0]: Applications Image 1 Start address • Bit [63:32]: Reserved
0xFF8	—	Reserved	

5.5. Generating Remote System Upgrade Image Files using Programming File Generator

Use the Intel Quartus Prime Programming File Generator tool to generate the Intel Stratix 10 remote system upgrade flash programming files.

5.5.1. Generating a Standard RSU Image

Follow these steps to generate a standard RSU image:

1. On the **File** menu, click **Programming File Generator**.
2. Select **Stratix 10** from the **Device family** drop-down list.
3. Select the configuration scheme from the **Configuration scheme** drop-down list. The current Intel Quartus Prime only supports remote system upgrade feature in **Active Serial x4**.
4. On the **Output Files** tab, assign the output directory and file name.
5. Select the output file type.

   Select the following file types for AS x4 configuration mode:

   - JTAG Indirect Configuration File (`.jic`)/Programmer Object File (`.pof`)
   - Memory Map File (`.map`)
   - Raw Programming File (`.rpd`)
6. On the **Input Files** tab, click **Add Bitstream**, select the factory and application image `.sof` files and click Open.
7. On the **Configuration Device** tab, click **Add Device**, select your flash memory and click **OK**. The Programming File Generator tool automatically populates the flash partitions.

(15) This image pointer has the highest priority for the initial standard remote system upgrade image.

8. Select the FACTORY_IMAGE partition and click **Edit**.
9. In the **Edit Partition** dialog box, select your factory image .sof file in the Input file drop-down list and click **OK**.

   Note: You must assign Page 0 to Factory Image. Intel recommends that you let the Intel Quartus Prime software assign the Start address of the FACTORY_IMAGE automatically by retaining the default value for **Address Mode** which is **Auto**. From the **Address Mode** drop down list, select **Block** to set an **End address** value for the FACTORY_IMAGE. The **Programming File Generator** reserves and assigns the start and end flash addresses to store BOOT_INFO, SPT0, SPT1, CPB0, and CPB1.
10. Select the flash memory and click **Add Partition**.
11. In the **Add Partition** dialog box, select for application image .sof file from the **Input file** drop-down list, assign the page number.
12. Repeat this step for additional application images and click **OK**. You can add up to three partitions for three application images. The **page 1** application image is the highest priority, and the **page 3** image is the lowest priority.
13. For .jic files,

    Click **Select** at the Flash loader, select your device family and device name, and click **OK**.
14. Click **Generate** to generate the remote system upgrade programming files. After generating the programming file, you can proceed to program the flash memory.

    Note: The generated .jic file contains only the initial flash data. If a remote host updates the initial flash image and then the application performs a verify operation, the verify operation fails. You can use the programmer to examine the flash content and compare it to the new flash image .rpd.

    Note: If you plan to update the factory image, Intel recommends reserving an additional 64 KB space for possible expansion of the factory image. Complete the following steps to reserve extra space for updates to the factory image:

    a. Identify the new end address by adding 64 KB to the existing END ADDRESS of the FACTORY_IMAGE. The end address is available in the .map file. For example, if the current end address is 0x00423FF, the new end address is 0x00433FF.
    b. Repeat the steps to regenerate the new .jic file. On the **Configuration Device** tab, select the FACTORY_IMAGE partition and click **Edit**. In the **Edit Partition** dialog box, under the **Address Mode** drop down list, select **Block** to set the new **End address** value for the FACTORY_IMAGE.

5.5.2. Generating a Single RSU Image

Follow these steps to generate single RSU image (.rpd) for adding or updating application image in user mode:

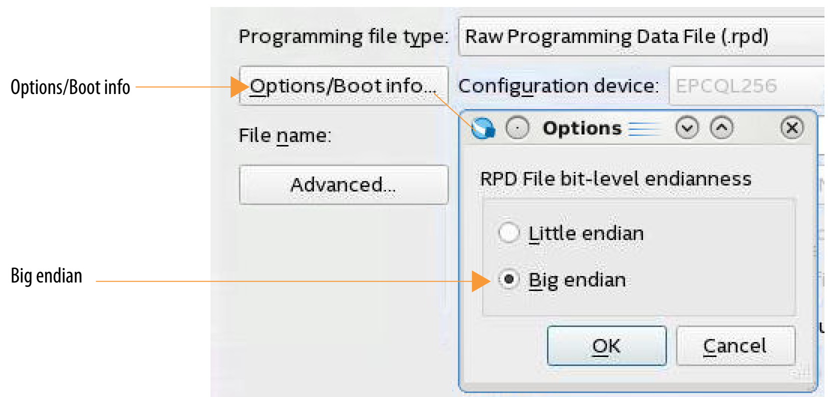
1. On the **File** menu, click **Programming File Generator**.
2. Select **Stratix 10** from the **Device family** drop-down list.
3. Select the configuration mode from the **Configuration mode** drop-down list. The current Intel Quartus Prime only supports remote system upgrade feature in **Active Serial x4**.

4. On the **Output Files** tab, assign the output directory and file name.
5. Select the output file type.

   Select the following file types for AS x4 configuration mode:

   - Raw Programming File (.rpd)
6. Click the **Edit...** button and assign the **Start address** for the image in flash memory.
7. By default, the .rpd file type is little-endian, if you are using a third-party programmer that does not support the little-endian format, click **Option/Boot Info** button. In the **Options** dialog box, set the RPD File Endianness to **Big Endian**.

Figure 41. Specifying RPD Bit-Level Endianness



8. On the **Input Files** tab, click **Add Bitstream**. Change the **Files of type** to SRAM Object File (*.sof). Then, select application image .sof file and click **Open**.
9. Click **Generate** to generate the remote system upgrade programming files. You can now program the flash memory.

5.6. Remote System Upgrade from FPGA Core Example

This section presents a complete remote system update example, including the following steps:

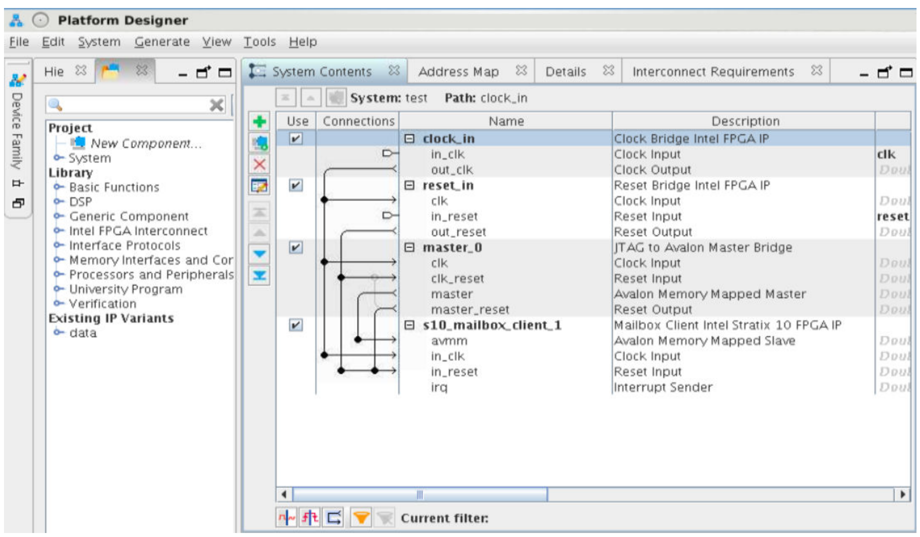
1. Creating the initial remote system update image (.jic) containing the bitstreams for the factory image and one application image.
2. Programming the flash memory with the initial remote system update image that subsequently configures the device.
3. Reconfiguring the device with an application or factory image.
4. Creating a single remote system update (.rpd) containing the bitstreams to add an application image in user mode.
5. Adding an application image.
6. Removing an application image.

5.7. Prerequisites

To run this remote system upgrade example, your system must meet the following hardware and software requirements:

- You should be running the Intel Quartus Prime Pro Edition software version 18.0 Update 1 or later.
- You should create and download this example to the Intel Stratix 10 SoC Development Kit.
- Your design should include the Mailbox Client Intel Stratix 10 FPGA IP that connects to a JTAG to Avalon Master Bridge as shown the Platform Designer system. The JTAG to Avalon Master Bridge acts as the remote system upgrade host controller for your factory and application images.

Figure 42. Required Communication and Host Components for the Remote System Update Design Example

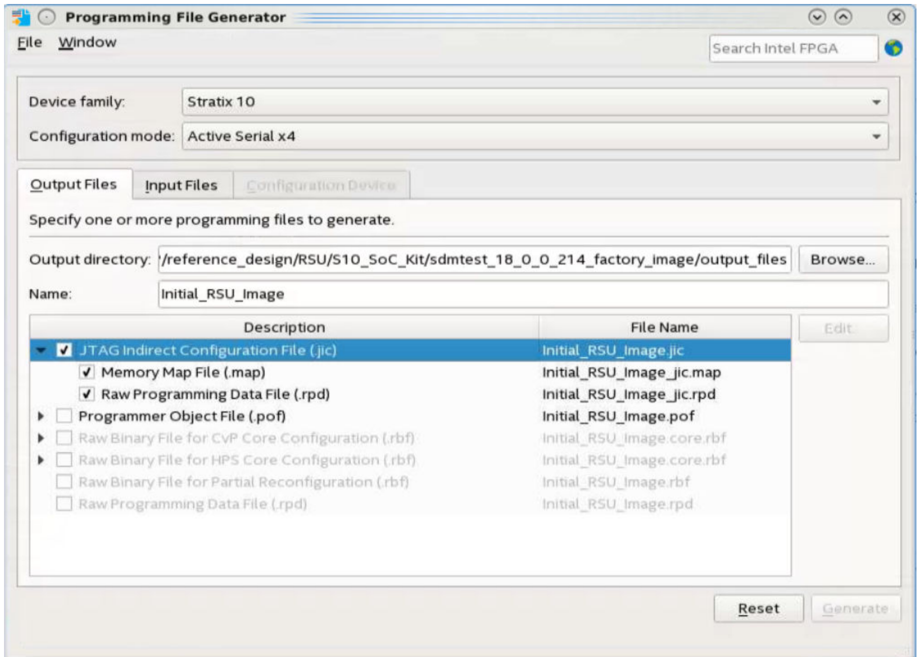


5.8. Creating Initial Flash Image Containing Bitstreams for Factory Image and One Application Image

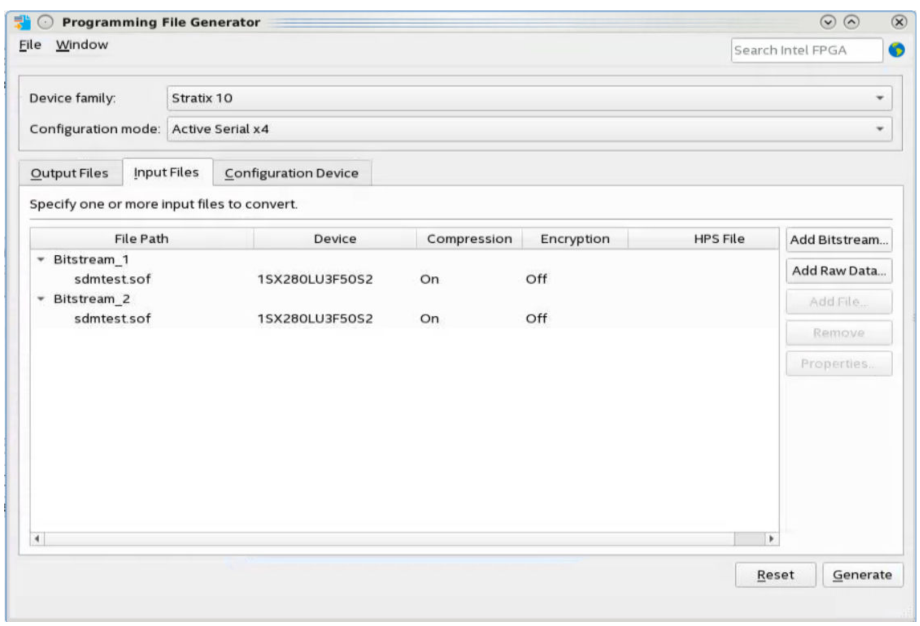
1. On the **File** menu, click **Programming File Generator**.
2. Select **Stratix 10** from **Device family** drop-down list.
3. Select the configuration mode from the **Configuration mode** drop-down list. The current Intel Quartus Prime only supports remote system upgrade feature in **Active Serial x4**.
4. On the **Output Files** tab, assign the output directory and file name.
5. Select the output file type.

   Select the following file types for Active Serial (AS) x4 configuration mode:

   - JTAG Indirect Configuration File (`.jic`)
   - Memory Map File (`.map`)
   - Raw Programming File (`.rpd`). It is optional to generate the `.rpd` file.



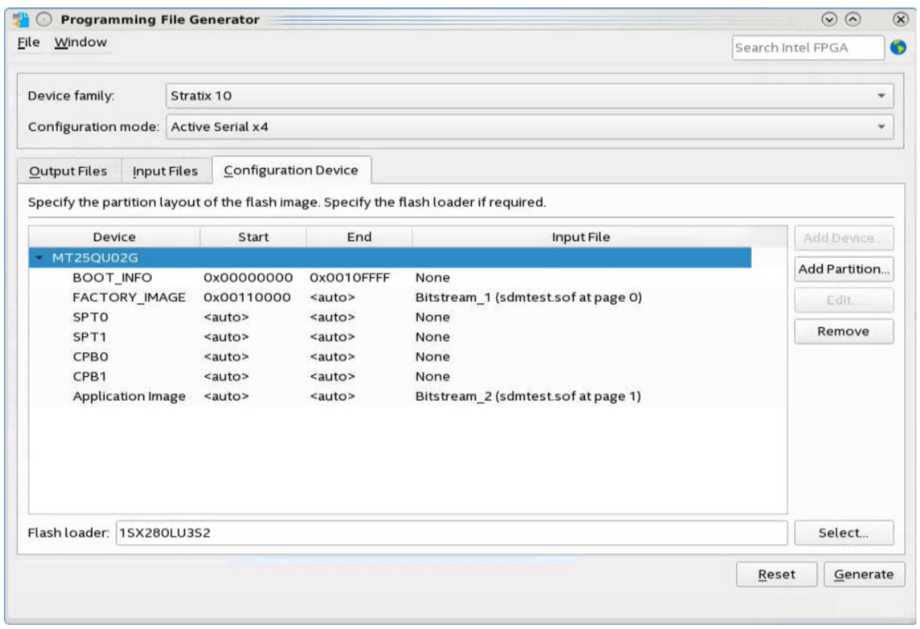
6. On the **Input Files** tab, click **Add Bitstream**, select the factory and application image `.sof` files and click **Open**.
   a. Bitstream_1 is the bitstream for factory image.
   b. Bitstream_2 is the bitstream for application image.



7. In the **Configuration Device** tab, click **Add Device**, select **MT25QU02G** flash memory and click **OK**. The Programming File Generator tool automatically populates the flash partitions.
8. Select the **FACTORY_IMAGE** partition and click **Edit**.

Send Feedback

9. On the **Edit Partition** dialog box, select **Bitstream_1** as the factory image `.sof` in the **Input file** drop-down list. Keep the default settings for **Page 0** and **Address Mode**. Click **OK**.
10. Select the **MT25QU02G** flash memory and click **Add Partition**.
11. In the **Add Partition** dialog box, select **Bitstream_2** for the application image `.sof` in the **Input file** drop-down list. Assign **Page: 1**.Keep the default settings for **Address Mode**. Click **OK**.
12. For **Flash loader** click **Select**. Select **Stratix 10** from **Device family** list. Select **1SX280LU3S2** for the **Device name**. Click **OK**.
13. Click **Generate** to generate the remote system upgrade programming files. The two following files are generated:
    a. `Initial_RSU_Image.jic`
    b. `Initial_RSU_Image_jic.map`



The following example output shows the generated `.map` file. The `.map` lists the start addresses of the factory image, CPB0, CPB1, and application image. The remote system update requires these addresses.

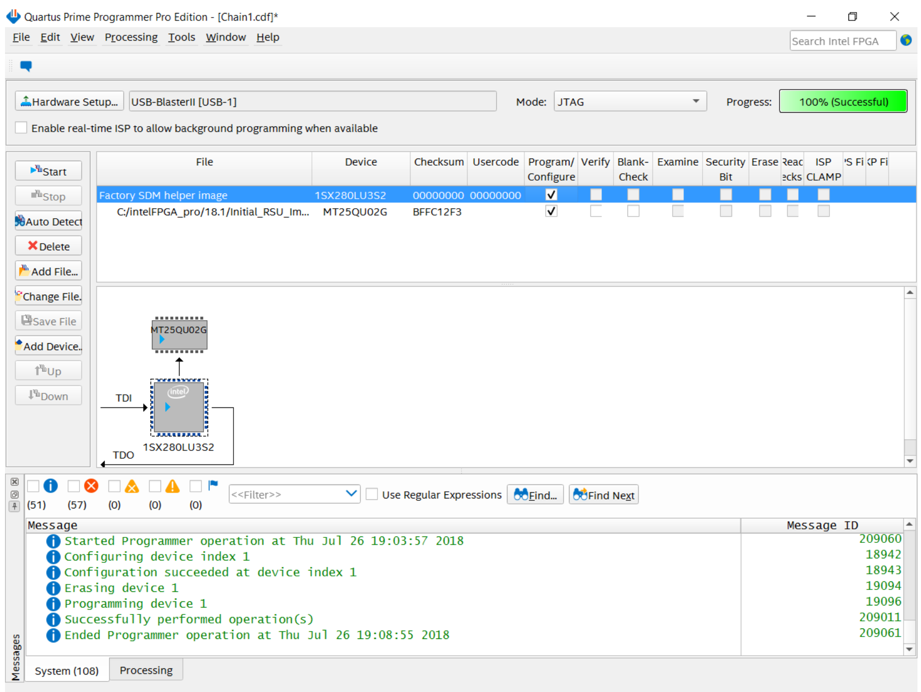
```
BLOCK                         START ADDRESS   END ADDRESS
BOOT_INFO                     0x00000000      0x0010FFFF
FACTORY_IMAGE                 0x00110000      0x002D3FFF
SPT0                          0x002D4000      0x002DBFFF
SPT1                          0x002DC000      0x002E3FFF
CPB0                          0x002E4000      0x002EBFFF
CPB1                          0x002EC000      0x002F3FFF
Application Image             0x002F4000      0x004B7FFF


Configuration device: 1SX280LU3S2
Configuration mode: Active Serial x4
Quad-Serial configuration device dummy clock cycle: 15
```

```
Notes:

- Data checksum for this conversion is 0xBFFB90A5

- All the addresses in this file are byte addresses
```

After generating the programming file, you can program the flash memory.

5.9. Programming Flash Memory with Initial Remote System Upgrade Image

1. Open **Programmer**, click **Add File**. Select the generated `.jic` file (`Initial_RSU_Image.jic`) and click **Open**.
2. Check the **Program/Configure** check box for the attached `.jic` file.
3. To begin programming the flash memory with the initial remote system upgrade image, click **Start**.
4. Configuration is complete when the progress bar reaches 100%. Power cycle the board to automatically configure the Intel Stratix 10 device with the application image using the AS x4 configuration scheme.



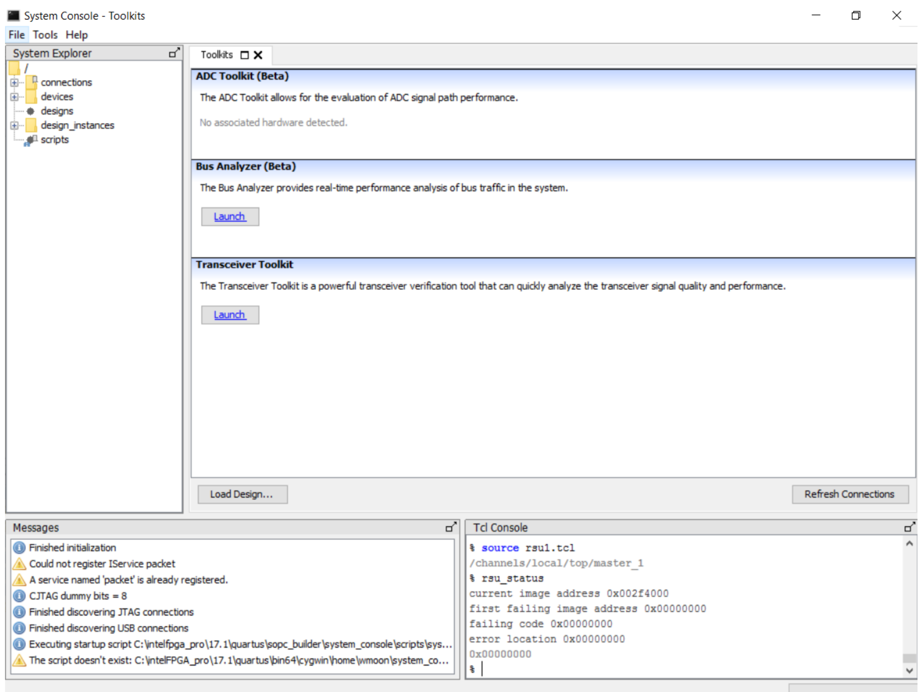
Note: This example does not assign the direct to factory pin. Consequently, the programmer configures the device with the application image. The Programming configures with the application image if the design does not use the direct to factory pin.

5. Use the RSU_STATUS command to determine which bitstream image the programmer is using as shown in the following example:

a. In the Intel Quartus Prime software, select **Tools ➤ System Debugging Tools ➤ System Console** to launch the system console.

b. In the Tcl Console pane, type `source rsu1.tcl` to open the example of Tcl script to perform the remote system upgrade commands.

c. Type the `rsu_status` command to report the current remote system upgrade status. You can retrieve the current running image address from the remote system upgrade status report. The current image address must match with the start address for the application image printed in the `.map` file, which indicates that the device is currently configured with the application image.



5.10. Reconfiguring the Device with an Application or Factory Image

The following steps describe the process to reconfigure the device with the desired application image or factory image using operation commands after the device is entering user mode.

1. The remote system upgrade host sends the RSU_IMAGE_UPDATE command to perform the remote system upgrade to the desired application image or factory image.

   a. For example, in the Tcl console of the system console, type the following commands to perform the remote system upgrade to the factory image and vice versa.

      i. `rsu_image_update 0x00110000`

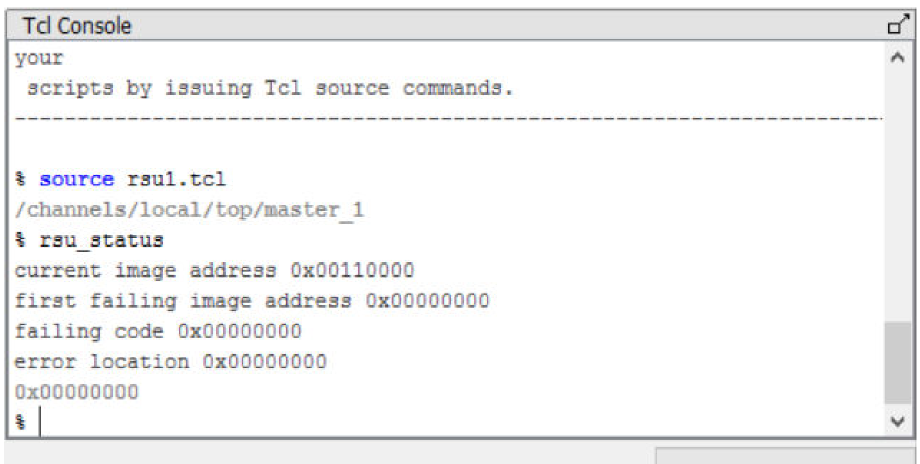
Note: The above is the command to reconfigure the device with factory image, `0x00110000` is the start address of the factory image shown in the `.map` file. The JTAG host connection via system console to the device disconnects once the device reconfiguration is successful. You must restart the system console to reestablish the connection with the device to perform next command.

ii. `rsu_image_update 0x002F4000`

Note: The above is the command to reconfigure the device with application image, `0x002F4000` is the start address of the application image shown in the `.map` file.

Optional: Retrieve the remote system upgrade status by using the `rsu_status` command to ensure you have successfully reconfigure the device with the desired image.

2. In the Tcl console of the system console, type `rsu_status` to retrieve the current image that is running in the device. The current image address must match with the start address for the factory or application image printed in the `.map` file depending on which image that reconfigured by the device. The following figure shows the device is being reconfigured with the factory image.



5.11. Adding an Application Image

Complete the following steps to add an application images to flash memory:

1. Set up exclusive access to the AS x4 interface and flash memory by running the `QSPI_OPEN` and `QSPI_SET_CS` commands. After running these commands, you have exclusive access to the AS x4 interface and flash until you relinquish access by running the `QSPI_CLOSE` command. Write the new application image to the flash memory using the `QSPI_WRITE` command.

2. Alternatively, the `rsu1.tcl` script includes the `program_flash` function that programs a new application image into flash memory. The following command accomplishes this task:

```
program_flash new_application_image.rpd 0x03FF0000 1024
```

The `program_flash` function takes three arguments:

a. The `.rpd` file to write to flash memory.

b. The start address.

c. Number of words to write for each `QSPI_WRITE` command. The `QSPI_WRITE` supports up to 1024 words per write instruction.

```
Tcl Console
% source rsu1.tcl
/channels/local/top/master_1
% program_flash new_application_image.rpd 0x03ff0000 1024
total number of words is 458752
total number of page is 448
total number of sector is 28
reading rpd is completed
start erasing flash
erasing flash is completed
start writing flash
writing flash is completed
```

3. Write the new application image start address to a new image pointer slot in the configuration firmware pointer block (CPB) using the `QSPI_WRITE` command. Ensure that the new image pointer slot value is `0xFFFFFFFF` before initiating the write.

   Note: You must update both copies (CBP0 and CBP1) when editing the configuration firmware pointer block and sub-partition table. Refer to Table 31 on page 84 for more details about the configuration firmware pointer block.

Based on the example described above, the address offset `0x20` in the CPB0 and CPB1 must point to the start address of the application image. The next new image pointer slot value must be `0xFFFFFFFF` before you write the start address of the new application image to the next image pointer slot.

Table 32. Configuration Firmware Point Block Contents

CPB Start Address + 0x20	Content	Value
CPB0 + 0x20 = 0x002E4020	Current application image pointer slot (highest priority)	0x002F4000
CPB0 + 0x28 = 0x002E4028	Next image pointer slot	0xFFFFFFFF
CPB1 + 0x20 = 0x002EC020	Current application image pointer slot (highest priority)	0x002F4000
CPB1 + 0x28 = 0x002EC028	Next image pointer slot	0xFFFFFFFF

You can use the `qspi_read` function verify that the new image pointer slot value is `0xFFFFFFFF` . The `qspi_read` function takes in two arguments:

1. Start address
2. Number of words to read

Figure 43. Verifying that the New Image Pointer Slot Value is 0xffffffff

```
Tcl Console
% qspi_read 0x002e4020 1
0x002f4000

% qspi_read 0x002e4028 1
0xffffffff

% qspi_read 0x002ec020 1
ISR is empty
0x002f4000

% qspi_read 0x002ec028 1
0xffffffff

%
```

You can now proceed to write the new application image address to next image slot by using the `qspi_write_one_word` function. The `qspi_write_one_word` function takes in two arguments:

1. Address
2. The value of the word

Figure 44. Writing an Address Pointer to the New Image Pointer Slot at 0x00234028

```
Tcl Console
% qspi_write_one_word 0x002e4028 0x03ff0000

% qspi_write_one_word 0x002ec028 0x03ff0000
```

You can now do a `qspi_read` function to the next image pointer slot to ensure that it is written with the start address of the desired new application image.

Verifying the Update to the New Image Pointer Slot at 0x00234028

```
Tcl Console
% qspi_read 0x002e4028 1
0x03ff0000

% qspi_read 0x002ec028 1
0x03ff0000
```

Host software can now reconfigure the Intel Stratix 10 FPGA with the new application image by asserting the `nCONFIG` pin. Alternatively, you can power cycle the PCB. After reconfiguration, check the current image address after the device reconfiguration. The expected address is `0x03ff0000`. By adding a new image, your application image list includes the newly added application image and the old application image, which is now a secondary image. The newly added application image has the highest priority.

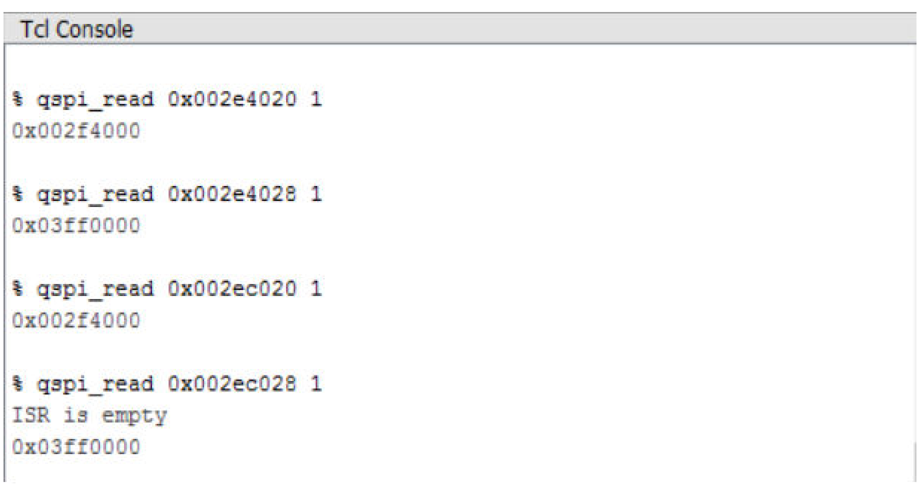
Note: When the remote system upgrade host loads an application image, the static firmware traverses the image pointer slots in reverse order. The new image has the highest priority when you restart the device.

5.12. Removing Application Image

1. Set up exclusive access to the AS x4 interface and flash memory by running the `QSPI_OPEN` and `QSPI_SET_CS` commands. After running these commands, you have exclusive access to the AS x4 interface and flash until you relinquish access by running the `QSPI_CLOSE` command. Write the new application image to the flash memory using the `QSPI_WRITE` command.
2. Write the application image start address stored in the image pointer slot of the configuration firmware pointer block (CPB0 and CPB1) to `0x00000000` by using the `QSPI_WRITE` command.

   Note: You must update both copies (copy0 and copy1) when editing the configuration firmware pointer block and sub-partition table.
3. Erase the application image content in the flash memory using the `QSPI_ERASE` command.
4. To remove a new application image, add another new application image in the next or subsequent image pointer slot or allow the device to fall back to the previous or secondary application image in your application image list. The following table shows correct entries for image pointer slots for CPB0 and CPB1 for offsets `0x20` and `0x28` :

CPB Start Address + 0x20	Content	Value
CPB0 + 0x20 = 0x002E4020	Old application image pointer slot (lower priority)	0x002F4000
CPB0 + 0x28 = 0x002E4028	Current/new application image pointer slot (highest priority)	0x03FF0000
CPB1 + 0x20 = 0x002EC020	Old application image pointer slot (lower priority)	0x002F4000
CPB1 + 0x28 = 0x002EC028	Current/New application image pointer slot (highest priority)	0x03FF0000


```
Tcl Console

$ qspi_read 0x002e4020 1
0x002f4000

$ qspi_read 0x002e4028 1
0x03ff0000

$ qspi_read 0x002ec020 1
0x002f4000

$ qspi_read 0x002ec028 1
ISR is empty
0x03ff0000
```

You can now remove the current or new application image address image pointer slot by writing the value to `0x00000000` using the `qspi_write_one_word` function as shown in the following example. The `qspi_write_one_word` function takes address and data arguments. Be sure to erase the application content that you just removed from flash memory.

Tcl Console

```
% qspi_write_one_word 0x002e4028 0x00000000

% qspi_write_one_word 0x002ec028 0x00000000
```

You can use a `qspi_read` to the image pointer slot at offset `0x28` for CBP0 and CPB1 to verify completion of the `qspi_write_one_word` commands .

Tcl Console

```
% qspi_read 0x002e4028 1
0x00000000

% qspi_read 0x002ec028 1
0x00000000
```

You can now configure the device with the old application image. The old application image has the highest priority if you power cycle the device or the host asserts the `nCONFIG` pin. You can run the `rsu_status` report to check the status of the current image address, `0x002f4000`.

6. Intel Stratix 10 Debugging Guide

6.1. Intel Stratix 10 Debugging Overview

Intel Stratix 10 devices employ a new configuration architecture. The Secure Device Manager (SDM), a dedicated hard processor, controls and monitors all aspects of device configuration from device power-on reset. This configuration architecture differs from previous Intel FPGA device families where state machines control configuration.

There are important differences between Intel Stratix 10 and previous device families with respect to available configuration modes, configuration pin behavior, and connection guidelines. In addition, the bitstream format is different. Knowing about these differences and how these pins behave can help you understand and debug configuration issues.

6.2. Configuration Pin Differences from Previous Device Families

Configuration Pin Names (Pre-Intel Stratix 10)	Intel Stratix 10 Pin Names	Notes
TRST	Not Available	Use the TMS reset sequence. Hold TMS high for 5 TCK cycles.
CLKUSR	OSC_CLK_1	An external source you can supply to increase the configuration throughput to 250 MHz. Using an external clock source Transceivers, the HPS, PCIe, and the High Bandwidth Memory (HBM2) require you this external clock. • 25 • 100 • 125 Refer to *Setting Configuration Clock Source* for instructions on setting the clock source and frequency in the Intel Quartus Prime Pro Edition software.
CRC_ERROR	Any unused SDM_IO (SEU_ERROR)	No dedicated location. Now called SEU_ERROR. Ignore until after CONF_DONE asserts.
CONF_DONE	SDM_IO5, SDM_IO16 (CONF_DONE)	No single dedicated pin location. No longer Open Drain. External pull-up Is not mandatory.
DCLK (PS - FPP)	AVST_CLK, AVSTx8_CLK	x8 mode has a dedicated clock input on SDM_IO14 (AVSTx8_CLK). For other Avalon-ST modes, use AVST_CLK. AVST_CLK and AVSTx8_CLK must be continuous and cannot pause during configuration.
DCLK (AS)	SDM_IO2 (AS_CLK)	When using the internal oscillator in AS mode, the AS_CLK runs in the range of 57 - 75 MHz. If you provide a 25 MHz, 100 MHz or 125 MHz clock to the OSC_CLK_1 pin, the AS_CLK can run up to 133 MHz.
DEV_OE	Not Available	

continued...

Configuration Pin Names (Pre-Intel Stratix 10)	Intel Stratix 10 Pin Names	Notes
DEV_CLRn	Not Available	
INIT_DONE	SDM_IO0 SDM_IO16 INIT_DONE	No longer Open Drain.
MSEL[0]	SDM_IO5 (MSEL[0])	After the SDM samples MSEL this pin functions as per the configuration mode selected. Do not connect directly to power. Use 4.7 KΩ pull-up or pull-downs, as appropriate.
MSEL[1]	SDM_IO7 (MSEL[1])	After the SDM samples MSEL, this pin functions as per the configuration mode selected. Do not connect directly to power. Use 4.7 KΩ pull-up or pull-downs, as appropriate.
MSEL[2]	SDM_IO9 (MSEL[2])	After the SDM samples MSEL, this pin functions as per the configuration mode selected. Do not connect directly to power. Use 4.7 KΩ pull-up or pull-downs, as appropriate.
NSTATUS	nSTATUS	No longer Open Drain. Intel recommends a 10 KΩ pull-up to V_{CCIO_SDM}.
NCE	Not Available	Multi-device configuration is not supported.
NCEO	Not Available	Multi-device configuration is not supported.
DATA[31:0] (PP32/PP16)	AVST_DATA[31:0]	Avalon-ST x8 uses SDM pins for data pins.
DATA[7:0] (PP8)	SDM _IO pins (AVSTx8_DATAn)	
nCSO[2:0]	SDMIO_8 (AS_nCSO3) SDMI_07 (AS_nCSO2) SDMI_09 (AS_nCSO1) SDM_IO5 (AS_nCSO0)	Intel Stratix 10 supports up to 4 cascaded AS devices
nIO_PULLUP	Not Available	Use a JTAG instruction to invoke.
AS_DATA0_ASDO	SDM_IO4 (AS_DATA0)	
AS_DATA[3:1]	SDM_IO6 (AS_DATA3) SDM_IO3 (AS_DATA2) SDM_IO1 (AS_DATA1)	Unlike earlier device families, the AS interface does not automatically tristate at power-on. When you set MSEL to JTAG, the SDM drives the AS_CLK, AS_DATA0-AS_DATA3, and AS_CS0-AS_CS3, MSEL pins until POR.
PR_REQUEST	GPIO*	No dedicated location.
PR_READY	GPIO*	No dedicated location.
PR_ERROR	GPIO*	No dedicated location.
PR_DONE	GPIO*	No dedicated location.
CVP_CONFDONE	Any unused SDM_IO CVP_CONFDONE	

Related Information

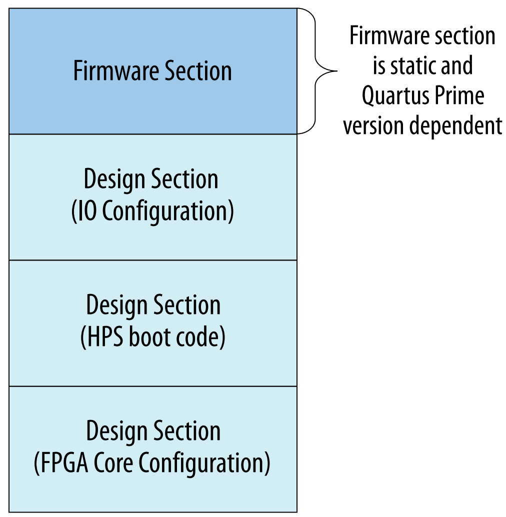
Setting Configuration Clock Source on page 23

6.3. Configuration File Format Differences

Detailed information about the configuration file format is proprietary. This topic explains the general structure and differences from previous device families.

The configuration file format differs significantly from previous device families. The configuration bitstream begins with a SDM firmware section. The SDM loads the boot ROM firmware during power-on reset. Design sections for I/O configuration, HPS boot code (if applicable), and fabric configuration follow the firmware section. Configuration begins after the SDM boot ROM performs device consistency checks.

Figure 45. Example of an Intel Stratix 10 Configuration Bitstream Structure



The firmware section is not part of the `.sof` file. The Intel Quartus Prime Pro Edition Programmer adds the firmware to the to the `.sof`. The programmer adds the firmware when configuring an Intel Stratix 10 device or when it converts the `.sof` to another format.

6.4. Understanding and Troubleshooting Configuration Pin Behavior

Configuration typically fails for one of the following reasons:

- The host times outs
- A configuration data error occurs
- An external event interrupts configuration
- An internal error occurs

Here are some very common causes of configuration failures:

- Check OSC_CLK_1 frequency. It must match the frequency you specified in the Intel Quartus Prime Software and the clock source on your board.
- Ensure a free running reference clock is present for designs using transceivers, PCIe, or HBM2.
- For designs using the HPS and the external memory interface (EMIF), ensure that the EMIF clock is present.
- For designs using SmartVID (-V devices), ensure that this feature is set-up and operating correctly. Ensure that the voltage regulator supports SmartVID.

Here are some debugging suggestions that apply to any configuration mode:

- To rule out issues with OSC_CLK_1 select the **Internal Oscillator** option in the Intel Quartus Prime.
- Try configuring the Intel Stratix 10 device with a simple design that does not contain any IP. If configuration via a non-JTAG scheme fails with a simple design, try JTAG configuration with the MSEL pins set specifically to JTAG.

The following topics describe the expected behavior of configuration pins. In addition, these topics provide some suggestions to assist in debugging configuration failures. Refer to the separate sections on each configuration scheme for debugging suggestions that pertain to a specific configuration scheme.

Related Information

- Debugging Guidelines for the Avalon-ST Configuration Scheme on page 33
- Debugging Guidelines for the AS Configuration Scheme on page 61
- Debugging Guidelines for the JTAG Configuration Scheme on page 67

6.4.1. nCONFIG

The nCONFIG pin is a dedicated, input pin in the SDM. nCONFIG has two functions:

- Hold-off initial configuration
- Initiate FPGA reconfiguration

The nCONFIG pin transition from low to high signals a configuration or reconfiguration request. The nSTATUS pin indicates device readiness to initiate FPGA configuration.

The configuration source can only change the state of the nCONFIG pin when it has the same value as nSTATUS. When the Intel Stratix 10 device is ready it drives nSTATUS to follow nCONFIG.

The host should assert nCONFIG to clear the device. Then the host should deassert nCONFIG initiate configuration. If nCONFIG asserts during a configuration cycle, that configuration cycle stops. The SDM expects a new configuration cycle to begin.

Debugging Suggestions

The host drives nCONFIG. Be sure that it is not floating or stuck low. nCONFIG should remain high during configuration.

6.4.2. nSTATUS

nSTATUS has the following two functions:

- To behave as an acknowledge for nCONFIG.
- To behave as an error status signal. It is important to monitor nSTATUS to identify configuration failures.

Note: nSTATUS does not go low for PR failures.

Generally, the Intel Stratix 10 device changes the value of nSTATUS to follow the value of nCONFIG, except after an error. For example, after POR, nSTATUS asserts after nCONFIG asserts. When the host drives nCONFIG high, the Intel Stratix 10 device drives nSTATUS high.

In previous device families the deassertion of nSTATUS indicates the device is ready for configuration. For Intel Stratix 10 devices, when using Avalon-ST configuration scheme, after the Intel Stratix 10 device drives nSTATUS high, you must also monitor the AVST_READY signal to determine when the device is ready to accept configuration data.

nSTATUS asserts if an error occurs during configuration. If an error occurs during configuration, the length of the nSTATUS low pulse varies depending upon the type of failure. The pulse ranges from .5 ms to 1.5 ms.

nSTATUS assertion is asynchronous to data error detection. Intel Stratix 10 devices do not support the **auto-restart configuration after error** option.

Previous device families implement the nSTATUS as an open drain with a weak internal pull-up. Consequently, you cannot wire OR an Intel Stratix 10 nSTATUS signal with the nSTATUS signal from earlier device families.

Debugging Suggestions

Ensure nSTATUS acknowledges nCONFIG. If nSTATUS is not following nCONFIG, the FPGA may not have exited POR. You may need to power cycle the PCB.

6.4.3. CONF_DONE and INIT_DONE

For Intel Stratix 10 devices, both CONF_DONE and INIT_DONE share multiplexed SDM_IO pins. Previous device families implement the CONF_DONE and INIT_DONE pins as open drains with a weak internal pull-up. Consequently, you cannot wire OR an Intel Stratix 10 CONF_DONE or INIT_DONE signal with the nSTATUS signal from previous device families. Otherwise, CONF_DONE and INIT_DONE behave as these signals behaved in earlier device families. If you assign CONF_DONE and INIT_DONE to SDM_IO16 and SDM_IO0, weak internal pull-downs pull these pins low at power-on reset. Ensure you specify these pins in the Intel Quartus Prime Software or in the Intel Quartus Prime settings file, (.qsf). CONF_DONE and INIT_DONE are low prior to and during configuration. CONF_DONE asserts when the device finishes receiving configuration data. INIT_DONE asserts when the device enters user mode.

Note: The entire device does not enter user mode at the same time.

CONF_DONE and INIT_DONE are optional signals. You can use these pins for other functions that the Intel Quartus Prime Pro Edition **Device and Pin Options** menu defines.

Debugging Suggestions

Place the CONF_DONE and INIT_DONE pins on the SDM_IO pins that correlate with the board-level connection. Refer to *SDM Pin Mapping* and *Setting Additional Configuration Pins* for more information.

Related Information

- SDM Pin Mapping on page 17
- Setting Additional Configuration Pins on page 21

6.4.4. SDM_IO Pins

Intel Stratix 10 devices include 17 SDM_IO pins that you can configure to implement specific functions such as CONF_DONE and INIT_DONE. The chosen function must follow the *Intel Stratix 10GX, MX, TX, and SX Device Family Pin Connections Guidelines*. The configuration bitstream controls the pin locations for the SDM_IO pins.

Internal Intel Stratix 10 circuitry pulls SDM_IO0, SDM_IO8 and SDM_IO16 weakly low through a 25 kΩ resistor. Internal Intel Stratix 10 circuitry pulls all SDM_IO pins weakly high during power-on.

Debugging Suggestions

Check the Intel Quartus Prime Pro Edition settings and Fitter report to ensure that the SDM_IO configuration matches your PCB design. The following screen shots show where to configure these signals and how to confirm the SDM_IO pin settings in the Fitter report.

Figure 46. Configuration Pin Selection in the Intel Quartus Prime Pro Edition Software

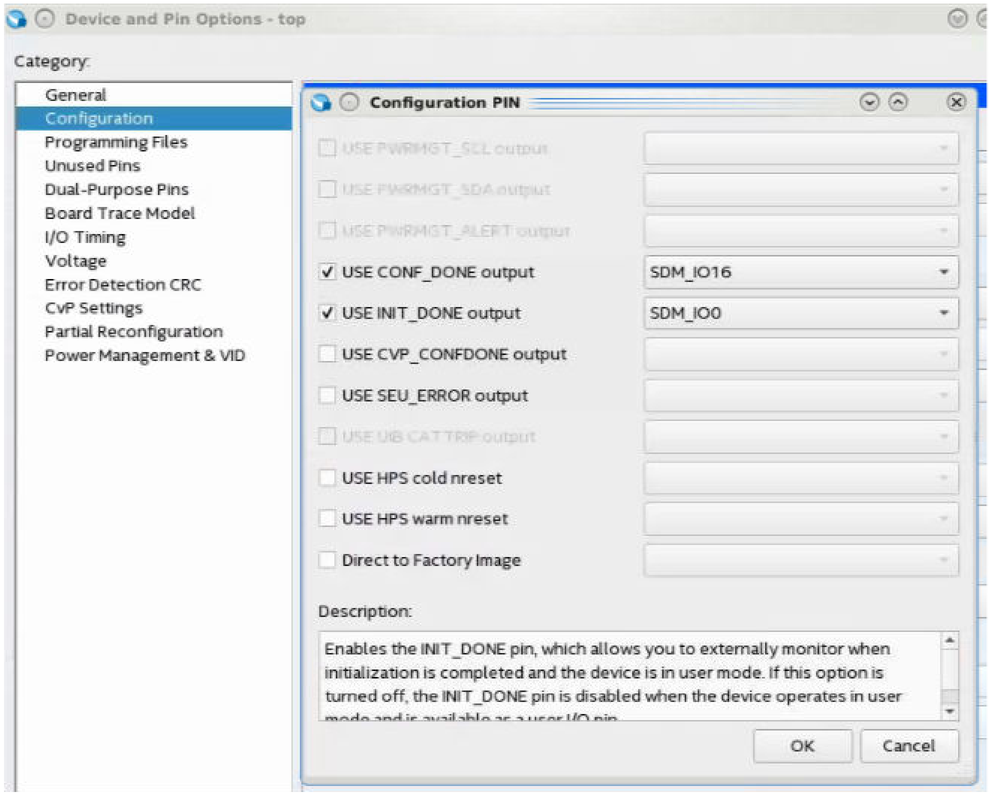
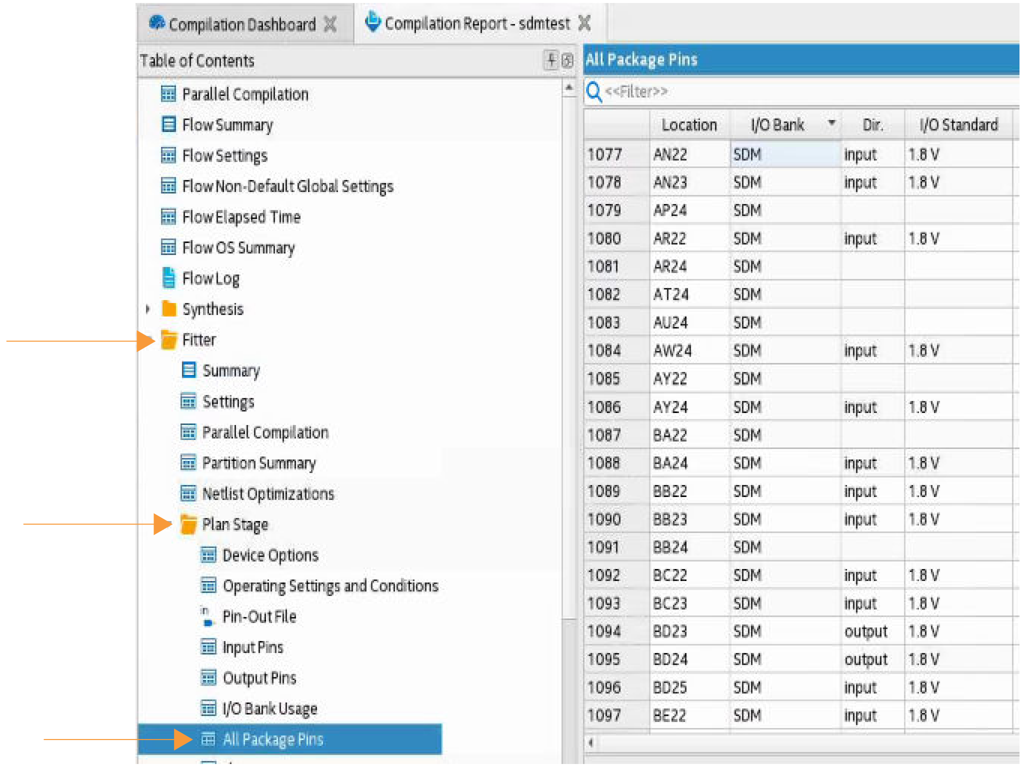


Figure 47. Fitter Report and SDM_IO Pin Reporting



Starting with the Intel Quartus Prime Pro Edition Software, version 18.1, an SDM debug tool is available through the System Console, **Tools ➤ System Debugging Tools ➤ System Console ➤ Stratix 10 SDM Debug**.

7. Intel Stratix 10 Configuration User Guide Archives

If an IP core version is not listed, the user guide for the previous IP core version applies.

IP Core Version	User Guide
18.0	Intel Stratix 10 Configuration User Guide
17.1	Intel Stratix 10 Configuration User Guide

8. Document Revision History for the Intel Stratix 10 Configuration User Guide

Document Version	Intel Quartus Prime Version	Changes
2019.01.25	18.1	Made the following changes: • Corrected misleading statement in the *Remote System Upgrade* topic. The Mailbox Client Intel Stratix 10 FPGA IP can instruct the SDM to reboot from the updated image. • Corrected statement 6 in the *Remote System Upgrade Configuration Sequence* topic: If loading the factory image fails, you can recover by reprogramming the QSPI flash with the RSU image using the JTAG interface. • Corrected encoding for MSEL for the Avalon-ST x16 configuration in Table 1 and Table 9. The correct encoding is 3'b101.
2018.11.02	18.1	Updated *Figure 39: Intel Stratix 10 Modules and Interfaces to Implement RSU Using Images Stored in Flash Memory* to exclude SD and MMC memory. These memory types are not supported in the current release.
2018.10.23	18.1	Added the following statement to the description of *Avalon-ST Configuration Timing* topic: *The AVST_READY signal is only valid when the nSTATUS pin is high.*
2018.10.10	18.1	Made the following changes: • Changed the number of remote system upgrade images supported from *more than 500* to *507* in *Remote System Upgrade Configuration Images*. • Updated the last two entries in the *Configuration Firmware Pointer Block Format* table.
2018.10.04	18.1	Made the following changes: • Corrected statement in the *Remote System Upgrade* topic. A command to the Mailbox Client Intel Stratix 10 FPGA Mailbox Client IP Core initiates reconfiguration. • Corrected the *Intel Stratix 10 Remote System Upgrade Components* figure and *Related Information* link. The mailbox component is the Mailbox Client Intel Stratix 10 FPGA IP Core.
2018.09.21	18.1	Made the following changes: • Added new chapter, *Remote System Upgrade* • Added new chapter, *Intel Stratix 10 Debugging Guide* • Added separate *Debugging Guidelines* topics in the Avalon-ST, AS, and JTAG configuration scheme sections. • Significantly expanded *Stratix 10 Configuration Overview Configuration Overview* chapter. • Added *Additional Clock and SmartVID Requirements for Transceivers, HPS, , High Bandwidth Memory (HBM2) and SmartVID* topic. • Expanded *OSC_CLK_1 Clock Input* topic to include additional usage requirements. • Added *AS Using Multiple Serial Flash Devices* topic. • Added numerous screenshots illustrating Intel Quartus Prime Pro Edition procedures.

continued...

Document Version	Intel Quartus Prime Version	Changes
		• Improved many figures illustrating configuration schemes. • Added the fact that you must have system administrator privileges to define a new flash device in the *Defining New CFI Flash Memory Device* topic. • Added MT28EW to the list of PFL II flash devices supported. • Moved almost all of the material describing the PFL II flash from an appendix to the *Intel Stratix 10 Configuration Schemes* chapter. • Edited entire document for clarity and style. • Corrected minor errors and typos.
2018.05.07	18.0	• Removed *Estimating the Active Serial Configuration Time* section. • Updated the OSC_CLK_1 supported frequency. • Added selecting flash loader step to *Generating Programming Files using Convert Programming Files*. • Added a note to TCK, TDI, TMS, and TDO stating that they are available for HPS JTAG chaining in SoC devices. • Removed instruction to drive nCONFIG low from POR in the following diagrams: — *Connections for AS x4 Single-Device Configuration* — *Connection Setup for AS Configuration with Multiple EPCQ-L Devices* — *Connection Setup for Programming the EPCQ-L Devices using the JTAG Interface* • Added a note in *OSC_CLK_1 Clock Input* stating that reference clocks to EMIF and PCIe IP cores must be stable and free running. • Removed .ekp file from *Overview of Intel Quartus Prime Supported Files and Tools for Configuration and Programming* figure. • Updated the *Configuring Intel Stratix 10 Devices using AS Configuration* section title to *Generating and Programming AS Configuration Programming Files*. • Updated *Configuration Schemes and Features Overview in Intel Stratix 10 Devices* table: — Added a note stating to contact sales representative for more information about support readiness. — Added a note stating to contact sales representative for more information about flash support other than EPCQ-L devices. • Removed NAND configuration support. • Updated *Configuration Sequence in Intel Stratix 10 Devices* figure by adding a looped flow arrow during Idle state. • Updated the MSEL note in *Intel Stratix 10 Device Configuration Pins* table. • Added a note to recommend OSC_CLK_1 for configuration clock source in *OSC_CLK_1 Clock Input*. • Updated CvP data width and maximum data rate in *Configuration Schemes and Features Overview in Intel Stratix 10 Devices* table. • Removed the multiple EPCQ-L configuration device support.

Date	Version	Changes
November 2017	2017.11.09	• Removed link to the *Configuration via Protocol (CvP) Implementation User Guide*. • Updated titles for *Device Security*, *Partial Reconfiguration*, and *Configuration via Protocol*.
November 2017	2017.11.06	• Updated *Option Bits Sector Format* table. • Updated a step in *Setting Additional Configuration Pins*. • Added *Converting .sof to .pof File* and *Programming CPLDs and Flash Memory Devices*. • Updated the .pof version value in *Storing Option Bits*.

continued...

Send Feedback

Date	Version	Changes
		• Added information about restoring start and end address for option bits in *Restoring Option Bit Start and End Address*. • Added note about pull-down resistor is recommended for `CONF_DONE` and `INIT_DONE` pins in *Additional Configuration Pin Functions*. • Added new subsection *Multiple EPCQ-L Devices Support*. • Added *Configuration Pins I/O Standard and Drive Strength* table. • Updated information about maximum additional data words when using 2-stage register synchronizer. • Updated the equation for minimum AS configuration time estimation. • Added *RBF Configuration File Format* section explaining the format of the .rbf file. • Updated *Configuration Sequence* to state that a firmware which is part of the configuration data if loaded in the device initially. • Updated description for **Number of flash devices used** parameter in the *PFL II Flash Interface Setting Parameters* table. • Updated *Configuration via Protocol* overview and added link to the Configuration via Protocol (CvP) Implementation User Guide. • Updated *Partial Reconfiguration* overview and added link to the *Creating a Partial Reconfiguration Design chapter of the Handbook Volume 1: Design and Compilation*. • Updated *Design Security Overview* descriptions. • Added note for Partial Reconfiguration feature and link to Partial Reconfiguration Solutions IP User Guide in *Intel Stratix 10 Configuration Overview*. • Removed SDM pin notes in *Intel Stratix 10 Configuration Overview*. • Updated internal oscillator's AS_CLK frequency in *Supported configuration clock source and AS_CLK Frequencies in Intel Stratix 10 Devices* table.
May 2017	2017.05.22	• Updated *Connection Setup for Programming the EPCQ-L Device using the AS Interface* figure. • Updated guideline to program the EPCQ-L device in *Programming EPCQ-L Devices using the Active Serial Interface*.
April 2017	2017.04.10	• Updated note for AS Fast Mode in *MSEL Settings for Each Configuration Scheme of Devices* table. • Added note to *Configuration via Protocol* recommending user to use AS x4 fast mode for CvP application. • Updated instances of Spansion to Cypress. • Added note to Normal Mode in *MSEL Settings for Each Configuration Scheme of Devices* table. • Updated note and description in *Configuration Overview*. • Removed AS x1 support. • Added *Connection Setup for SD/MMC Single-Device Configuration* figure. • Updated *Connections for AS x4 Single-Device Configuration*, *Connection Setup for AS Configuration with Multiple EPCQ-L Devices*, *Connection Setup for Programming the EPCQ-L Devices using the JTAG Interface*, *Connection Setup for NAND Flash Single-Device Configuration*, and *Connection Setup for SD/MMC Single-Device Configuration* to include note about nCONFIG test point. • Added note in *Avalon-ST Configuration* stating that AVST_CLK should be continuous.
February 2017	2017.02.13	• Updated *Configuring Stratix 10 Devices using AS Configuration* section and subsections to include `.jic` for AS configuration scheme. • Added *Programming .jic files into EPCQ-L Device*. • Updated the SDM description. • Updated SDM block diagram by adding Mailbox block and note for Avalon-ST x8 configuration scheme.

continued...

Date	Version	Changes
		• Updated Configuration Sequence Diagram. • Updated configuration sequence descriptions. • Updated *Avalon-ST Bus Timing Waveform* figure. • Added note to Avalon-ST in *Stratix 10 Configuration Overview* table. • Updated ASx4 max data rate in *Stratix 10 Configuration Overview* table. • Removed *Configurable Node* subsection.
December 2016	2016.12.09	• Updated max data rate for ASx1. • Updated the *Configuration Sequence in Stratix 10 Devices* figure. • Updated configuration sequence description. • Added JTAG configuration sequence description. • Added Parallel Flash Loader II IP core.
October 2016	2016.10.31	Initial release

1. Supported CFI Flash Memory Devices

Table 33. CFI Flash Memory Devices Supported by PFL II IP Core

Manufacturer	Product Family	Data Width	Density (Megabit)	Device Name(16)
Micron	C3	16	8	28F800C3
			16	28F160C3
			32	28F320C3
			64	28F640C3
	J3	8 or 16	32	28F320J3
			64	28F640J3
			128	28F128J3
		16	256	JS29F256J3
	P30	16	64	28F640P30
			128	28F128P30
			256	28F256P30
			512	28F512P30
			1000	28F00AP30(17)
			2000	28F00BP30
	P33	16	64	28F640P33
			128	28F128P33
			256	28F256P33
			512	28F512P33
			1000	28F00AP33
			2000	28F00BP33
	MT28EW01GABA1	8 or 16	1024	MT28EW01GABA1
	MT23FW02	16	2048	MT23FW02
	M29EW	8 or 16	256	28F256M29EW
			512	28F512M29EW
			1000	28F00AM29EW

continued...

(16) The PFL II IP core supports top and bottom boot block of the flash memory devices. For Micron flash memory devices, the PFL II IP core supports top, bottom, and symmetrical blocks of flash memory devices.

Manufacturer	Product Family	Data Width	Density (Megabit)	Device Name[16]
	M29W	8 or 16	16	M28W160CT
				M28W160CB
				M29W160F7
				M29W160FB
			32	M29W320E
				M29W320FT
				M29W320FB
			64	M29W640F
				M29W640G
			128	M29W128G
			256	M29W256G
	M29DW	8 or 16	32	M29DW323DT
				M29DW323DB
	G18	16	512	MT28GU512AAA1EGC-0SIT
			1024	MT28GU01GAAA1EGC-0SIT [17]
	M58BW	32	16	M58BW16FT
				M58BW16FB
			32	M58BW32FT
		16 or 32	32	M58BW32FB
Cypress	GL-P[18]	8 or 16	128	S29GL128P
			256	S29GL256P
			512	S29GL512P
			1024	S29GL01GP
	AL-D	8 or 16	16	S29AL016D
			32	S29AL032D
	AL-J	8 or 16	16	S29AL016J
	AL-M	8 or 16	16	S29AL016M
	JL-H	8 or 16	32	S29JL032H
			64	S29JL064H
	WS-N	16	128	S29WS128N

continued...

(16) The PFL II IP core supports top and bottom boot block of the flash memory devices. For Micron flash memory devices, the PFL II IP core supports top, bottom, and symmetrical blocks of flash memory devices.

Manufacturer	Product Family	Data Width	Density (Megabit)	Device Name[16]
	GL-S	16	128	S29GL128S
			256	S29GL256S
			512	S29GL512S
			1024	S29GL01GS
Macronix	MX29LV	16	16	MX29LV160D
			32	MX29LV320D
			64	MX29LV640D
				MX29LV640E
	MX29GL	16	128	MX29GL128E
			256	MX29GL256E
Eon Silicon Solution	EN29LV	16	16	EN29LV160B
	EN29GL	16	32	EN29LV320B
			128	EN29GL128

(16) The PFL II IP core supports top and bottom boot block of the flash memory devices. For Micron flash memory devices, the PFL II IP core supports top, bottom, and symmetrical blocks of flash memory devices.

(17) Intel tested flash device.

(18) Supports page mode.