LatticeECP3 Slave SPI Port User Guide

August 2019 **Technical Note FPGA-TN-02136**

Introduction

Prior to the introduction of the Serial Peripheral Interface Bus (SPI), the standard methods for configuring an FPGA using a CPU were through the following ports or interfaces:

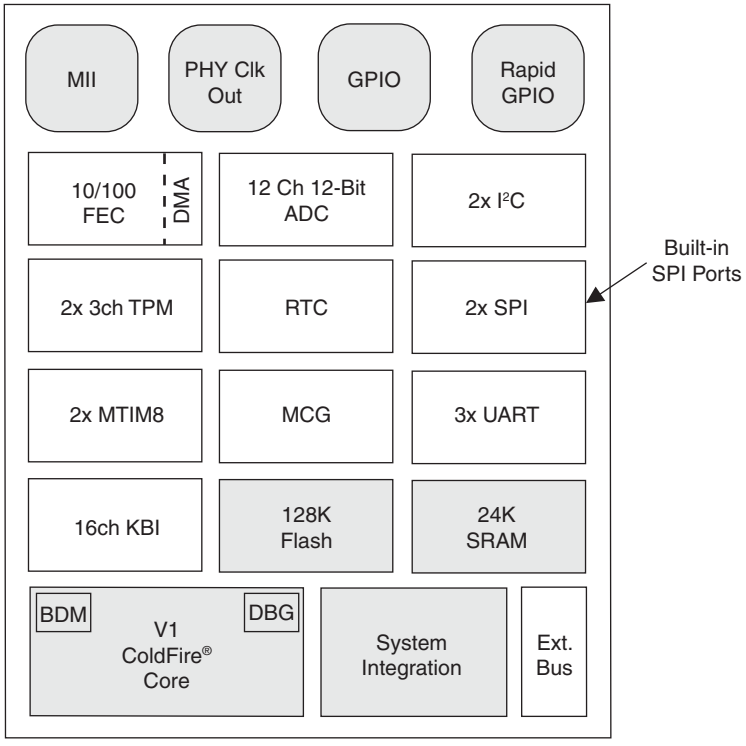
- JTAG
- SCM (Serial Configuration Mode)
- PCM (Parallel Configuration Mode)

Each port has advantages and disadvantages depending on the given application:

- The JTAG port requires a custom driver, which can be challenging to integrate into the system software.
- The SCM port does not support read back.
- The PCM port requires usage of approximately 14 General Purpose Input/Output pins (GPIO). GPIOs are precious resources that may be needed for user I/O.
- For most of the FPGA devices in the market place, read back is only supported with a PCM (CPU type) port.

SPI is an industry standard interface that is available on most CPUs and serial Flash memory devices. The 32-bit CPU from Freescale™ Semiconductor shown in Figure 1 illustrates the availability of the SPI interface. The drivers for reading and writing from SPI memory devices are readily available for modern digital systems.

Figure 1. Example CPU with Built-in SPI Port



Freescale Semiconductor
MCF51CN128 Series Block Diagram

© 2019 Lattice Semiconductor Corp. All Lattice trademarks, registered trademarks, patents, and disclaimers are as listed at www.latticesemi.com/legal. All other brand or product names are trademarks or registered trademarks of their respective holders. The specifications and information herein are subject to change without notice.

The advantages of the SPI port are as follows.

1. Most CPUs have built-in SPI interfaces, thus from the system designer's perspective:

- The SPI interface is free.
- The driver is already built-in for easy and seamless system integration.
- The programming data file is a separate bitstream file that can be stored in system memory.

2. The SPI clock speed tracks the system clock, thus from the FPGA configuration's perspective:

- The SPI port is ~10 times faster than using GPIO pins.
- The configuration time is ~10 times faster than GPIO pins.

3. The SPI interface supports read back.

- Allows for real time device monitoring.
- Supports configuration debugging and diagnostic.
- Requires only four (4) pins compared to fourteen (14) pins for the PCM interface.

Definitions

Bitstream

The Bitstream Data File (.bit file) is the configuration data file in the format that can be written directly into the FPGA devices to configure the SRAM cells. The file is expressed in binary hex format.

Erase

The process of clearing all the SRAM cells state to a logical zero (0).

Configure

The process of writing the bitstream pattern into the SRAM cells.

Refresh

The process of re-triggering a bitstream write operation. It is activated by toggling of the PROGRAMN pin or issuing a REFRESH command, which emulates the PROGRAMN pin toggling. Only the JTAG port and the Slave SPI port support the REFRESH command.

Direct Mode (Foreground Mode)

The device is in a configuration mode and all the I/O pins are kept tri-stated.

Background Mode

The device is in a configuration mode where all the I/O pins remain operational.

User Mode

The device is not in a configuration mode. The device is configured and operational.

SPI

Serial Peripheral Interface Bus. An industry standard, full duplex, synchronous serial data link that uses a four wire interface. The interface supports a single master and single or multiple slaves.

Master SPI

A configuration mode where the FPGA drives the master clock and issues commands to read the bitstream from an external SPI Flash device.

Slave SPI (SSPI)

A configuration mode where the CPU drives the clock and issues commands to the FPGA for writing the bitstream into the SRAM cells.

Secure

To protect the SRAM cells from reading.

Advanced Security

The advanced features provide additional security to the device. Examples are Encryption and the Dual Boot Feature.

Dual Boot

The device can boot from two patterns, the Primary pattern and the Golden pattern, both residing in the external SPI Flash device. If the Primary pattern is corrupted or otherwise fails to load, the device will boot from the Golden pattern. This feature is available only in SPIm configuration mode.

Primary Boot

In Dual Boot Mode, the FPGA device will load this pattern in first. Only one Primary pattern is allowed.

Golden Boot

In Dual Boot Mode, the Golden Boot pattern is loaded when the Primary Boot fails. Only one Golden boot pattern is allowed.

Multiplex Formation

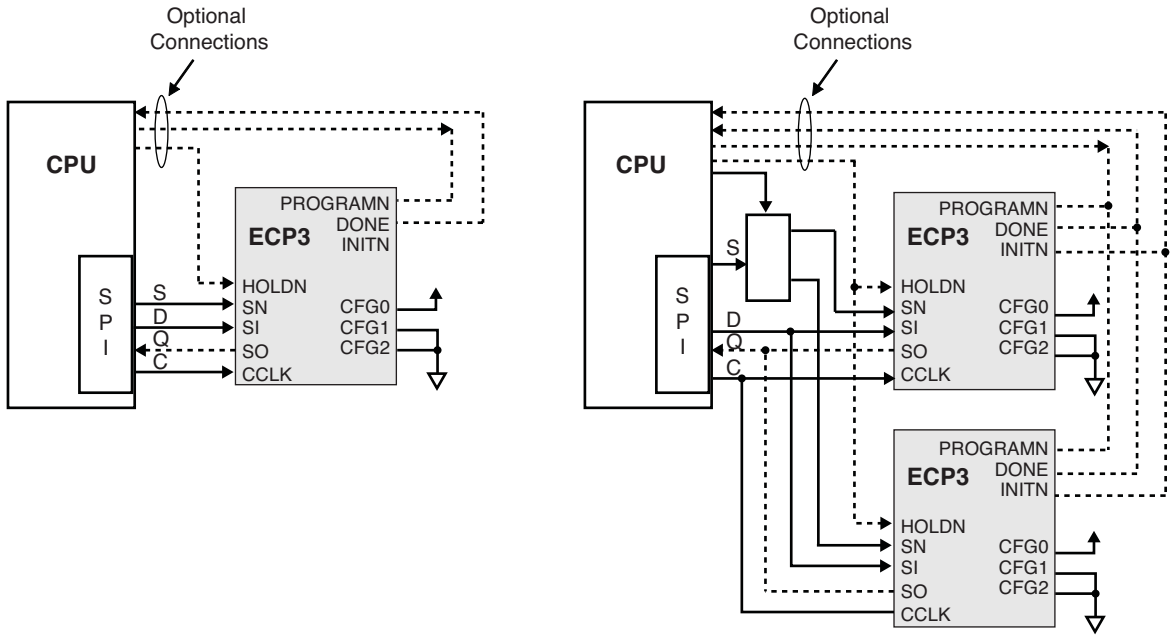
This is the method to connect multiple FPGA devices together by using the Chip Select pin of the Slave SPI port to select one device at a time for Configuration. The SPI standard does not support serial daisy chain formation. Please see the Advanced Application section for using an external SPI Flash device to form the daisy chain in conjunction with the Slave SPI port.

Purpose

This application note provides the details for using the built-in SPI port in the LatticeECP3™ devices for the following operations:

1. Configuration: Writing the bitstream into the device.
2. Read Back: Read the status register, Usercode, or bitstream from the device.
3. Programming: Program the bitstream into an external SPI Flash device.

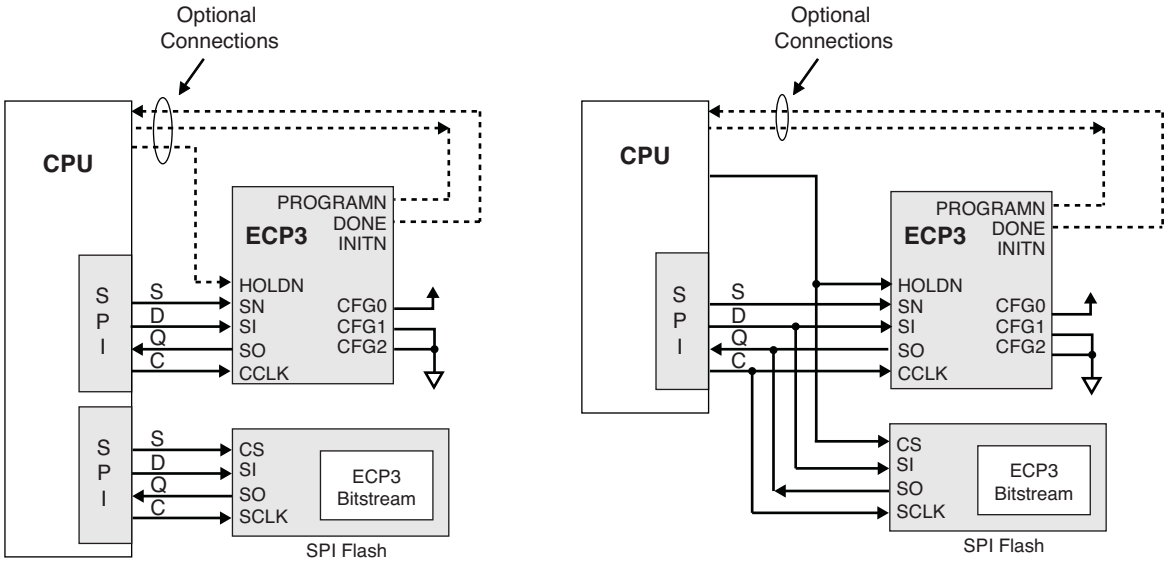
Figure 2. LatticeECP3 Slave SPI Port with CPU and Single or Multiple Devices



Notes:

* The dotted lines indicate optional connections.
* The wake up time of the device does vary with the bitstream size and the speed of the SPI port. Lattice recommends connecting the DONE pin to the CPU to monitor when the configuration is complete.
* If the bitstream for the two LatticeECP3 devices is the same, the chip select logic (S/SN) is not required.
* The SO to Q connection is optional if read back is not needed and the DONE pin is connected.

Figure 3. LatticeECP3 Slave SPI Port with SPI Flash



Notes:

* The dotted lines indicate the connection is optional.
* The LatticeECP3 bitstream can reside in the SPI Flash device instead of the system Flash memory. The advantage of this is that the bitstream can be easily updated without changing the system software.

Slave SPI Port Description

Characteristics of the Slave SPI Port

The Slave SPI port is considered an intelligent port. It is capable of performing read and write actions based on the command shifted into the device. When the LatticeECP3 is in Background Mode, only read type commands are supported, allowing for device verification or debugging.

A device is considered a Slave SPI device if its clock is driven from an external device. A Master SPI device drives the clock out to the SPI slaves.

The Slave SPI port only supports a single device. It does not directly support serial daisy chains. See the Advanced Application section for using an external SPI Flash device to form a daisy chain in conjunction with the Slave SPI port.

The Slave SPI port reads data from the data input pin (SI) on the rising edge of the clock. The port transfer data out to the data output pin (SO) on the falling edge of the clock.

The command set consists of 8-bit opcodes. Some of the commands have a 24-bit operand following the 8-bit opcode. The Slave SPI port is a byte bounded port; all input and output data must be byte bounded.

Method to Enable the Slave SPI Port

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

1. Setting the CFG[2:0] pin to [0,0,1].

   When the device is powered up, or when the PROGRAMN pin is toggled, the device checks the state of the CFG pins. If they are set to [0,0,1], then the device will choose the Slave SPI port as the configuration port. This is the only method to enable the Slave SPI port as the configuration port. A port is said to be a configuration port when it is capable of executing both bitstream write and read commands.

2. Enabling Slave SPI persistence.

   The configuration bitstream contains an optional Slave SPI persistence bit. When the device completes configuration and wakes up, it checks the persistent bit to determine if the Slave SPI port is to remain operational once in User Mode. This selection is independent of the CFG pins setting. A Slave SPI (SSPI) port enabled by persistence is only capable of read commands and PROGRAM_SPI0 command for SPI bridging.

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

Slave SPI Port Pin Definitions

By definition, the SPI port is a four (4) wire port. The HOLDN pin was added by SPI Flash vendors to provide the CPU a method to support suspension. The LatticeECP3 devices also support the HOLDN pin to provide the CPU a method to suspend data transmission when it cannot process the large bitstream in one single burst and needs time to fetch the bitstream in fragments.

CSN/SN

This is an active low chip select pin. It serves two important functions.

1. High to low transition: Resets the device, prepares it to receive commands.

2. Low to high transition: Completes or terminates the current command.

Note that when the SN pin is driven from low to high prior to the completion of shifting in an 8-bit command, the command is considered ‘under shifted’. The LatticeECP3 devices ignore the command in the command buffer when under shifting happens. Driving SN low again will reset the command buffer in preparation for the next command.

CCLK

This is the clock input pin.

D[3]/SI

This is the serial input pin for commands, operands, and bitstream data.

D[4]/SO

This is the serial output pin for read back data. It is normally tri-stated with an internal pull-up. It becomes active only when the command is a read type command.

CSN1/HOLDN

Although not a standard SPI pin, this is an industry standard pin provided to allow the CPU to suspend data transmission. This pin can be asserted while shifting the bitstream into the device.

Do not assert the HOLDN pin while shifting commands or operands into the device.

Specifications and Timing Diagrams

Slave SPI Port Waveforms

Data and commands shift into the SI pin on the rising edge of clock. Data is shifted out of the SO pin on the falling edge of clock. Only a read command will cause the SO pin to be enabled for data read out.

The Slave SPI read and write waveforms are shown in Figure 4 and Figure 5. The Slave SPI HOLDN pin waveform is shown in Figure 6.

Figure 4. Slave SPI Read Waveforms

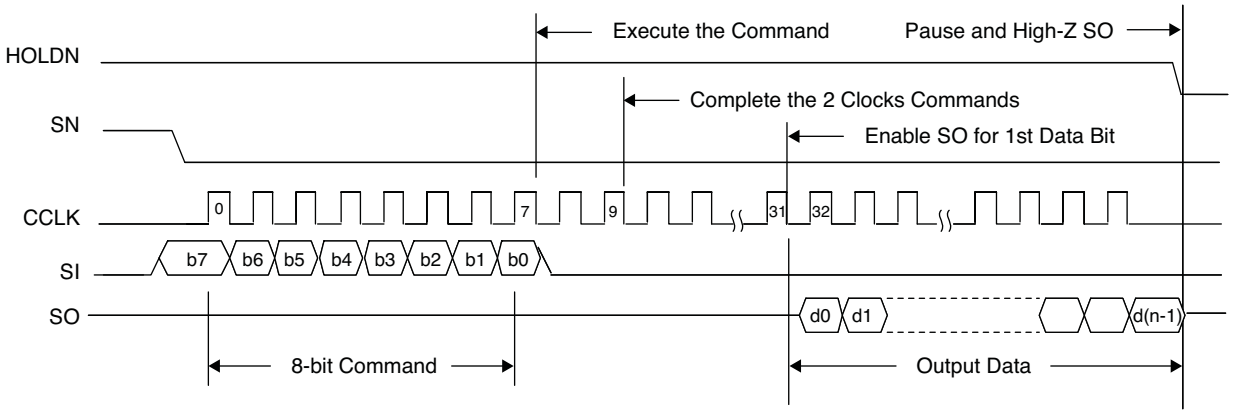
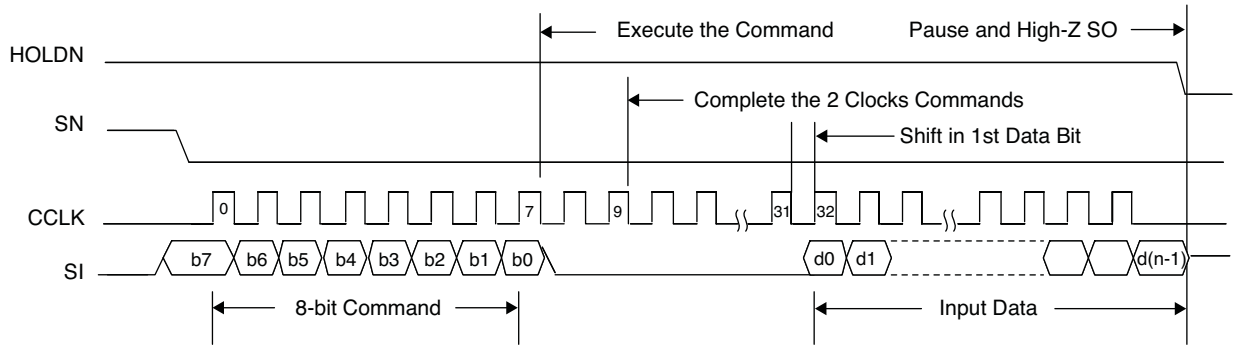
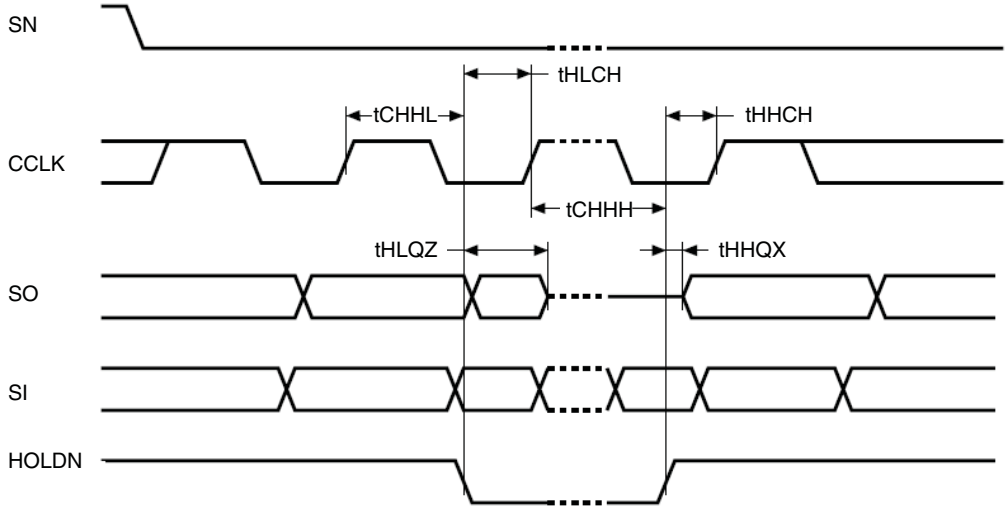


Figure 5. Slave SPI Write Waveforms



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

Figure 6. Slave SPI HOLDN Waveforms



Slave SPI Port AC Timing Requirements

The Slave SPI port AC timing requirements are listed in Table 1.

Table 1. AC Timing Requirements

Description	Parameter	Min.	Max.	Units
CCLK Frequency	f_{CCLK}		33	MHz
CCLK Minimum High Pulse	t_{SSCH}	5		ns
CCLK Minimum Low Pulse	t_{SSCL}	5		ns
HOLDN Low Setup Time (relative to CCLK)	t_{HLCH}	5		ns
HOLDN Low Hold Time (relative to CCLK)	t_{CHHH}	5		ns
HOLDN High Hold Time (relative to CCLK)	t_{CHHL}	5		ns
HOLDN High Setup Time (relative to CCLK)	t_{HHCH}	5		ns
HOLDN to Output High-Z	t_{HLQZ}		9	ns
HOLDN to Output Low-Z	t_{HHQX}		9	ns

LatticeECP3 Additional Characteristics

- Do not drive the Chip Select (SN) pin from low to high to suspend an operation. Doing so will terminate the current command, since the low to high transition will clear the command buffer.

Solution: Use the HOLDN pin, or halt and hold the clock pin low to pause.

Note: When shifting in commands, the only method to suspend clocking is by halting the clock.

- The bitstream is one continuous data record. There is no concept of page size or sector size or address or address offset. When clocking the bitstream into the device, the Chip Select (SN) pin must be kept low until the entire bitstream is clocked into the device.

Solution:If pausing is necessary, use the HOLDN pin, or halt and hold the clock pin low.

- If it is necessary to suspend a read back operation, do not drive the Chip Select high.

Solution:If pausing is necessary, use the HOLDN pin, or halt and hold the clock pin low.

- If it is necessary to use an encrypted bitstream, the Encryption Key must first be programmed into the device. The Slave SPI port does not support Encryption Key programming. Only the JTAG port supports Encryption Key programming.

Solution: Use ispVM or a desk top programmer to program the Encryption Key into the device.

Table 2. Slave SPI Command Table

Command	Binary	Hex	Operand[1]	Description
READ_INC	b00000001	h01	24 bits	Read out the configuration bit stream.
READ_USERCODE	b00000011	h03	24 bits	Read the 32-bit USERCODE.
READ_CONTROL	b00000100	h04	24 bits	Read out the content of the control register.
READ_ID	b00000111	h07	24 bits	Read out the 32-bit JTAG IDCODE of the device.
READ_STATUS	b00001001	h09	24 bits	Read out the status register for the state of INITN and DONE bit.
CLEAR	b01110000	h70	24 bits	Clear the configuration RAM and control register 0.
WRITE_INC	b01000001	h41	24 bits	Connect SI to DI (bitstream input)
WRITE_EN	b01001010	h4A	24 bits	Enable the configuration operation.
REFRESH	b01110001	h71	24 bits	Same as toggling the program pin for clear all and activate the configuration.
WRITE_DIS	b01001111	h4F	24 bits	Disable the configuration operation.
PROGRAM_SPI0	b01110100	h74	24 bits	Connect SPI Host Port to the SPI0 interface to program the SPI Flash.

1. Operands are dummy clocks to provide extra timing for the device to execute the command. The data presented at the SI pin during these dummy clocks can be 0x000.

Table 3. Slave SPI Command Usage Table

Command	OPCODE	Class	Flow Operation Number	Delay Time
READ_INC	h01	A		None
READ_USERCODE	h03	A		None
READ_CONTROL	h04	A		None
READ_ID	h07	A	2, 3	None
READ_STATUS	h09	A	8	None
CLEAR	h70	D		[1]
WRITE_INC	h41	B	6, 7	None
WRITE_EN	h4A	C	5	None
REFRESH	h71	D		[1]
WRITE_DIS	h4F	C	9	None
PROGRAM_SPI0	h74	E		None

1. Delay time depends on the bitstream size and clock frequency. Duration could be in seconds.

Device Status Register

The LatticeECP3 has a 32-bit device status register, which indicates the status of the device. The READ_STATUS command shifts out this 32-bit internal status of the device. The first bit shifted out on the SO pin is bit 0, and the last bit is bit 31.

Only the bits that are relevant to the Slave SPI Port are listed.

Table 4. 32-Bit Device Status Register

Bit	Function	Status Value			Comments
		Reset Value	0	1	Bit Value Description
0	CRC Error	0	Pass	Fail	Bitstream is corrupted.
2	Bitstream Invalid Command	0	Pass	Fail	Bitstream is corrupted.
4	Encryption Key Locked	0/1	No	Yes	Encryption Key programmed.
5	Encrypt Bitstream Valid	0	No	Yes	Encrypt Key matches Encrypt bitstream.
6	Alignment Preamble Found	0	No	Yes	The encrypted bitstream is valid.
7	Encryption Preamble Found	0	No	Yes	The bitstream is an encrypted bitstream.
8	Standard Preamble Found	0	No	Yes	Detected the preamble code 0xBDB3.
15	Memory Cleared	0	No	Yes	SRAM was cleared.
16	Device Secured	0	No	Yes	Read back is disabled.
17	Done	0	No	Yes	Done pin is High.
Others	Reserved	x	x	x	Unused/Undefined.

Slave SPI Configuration Flow Diagrams

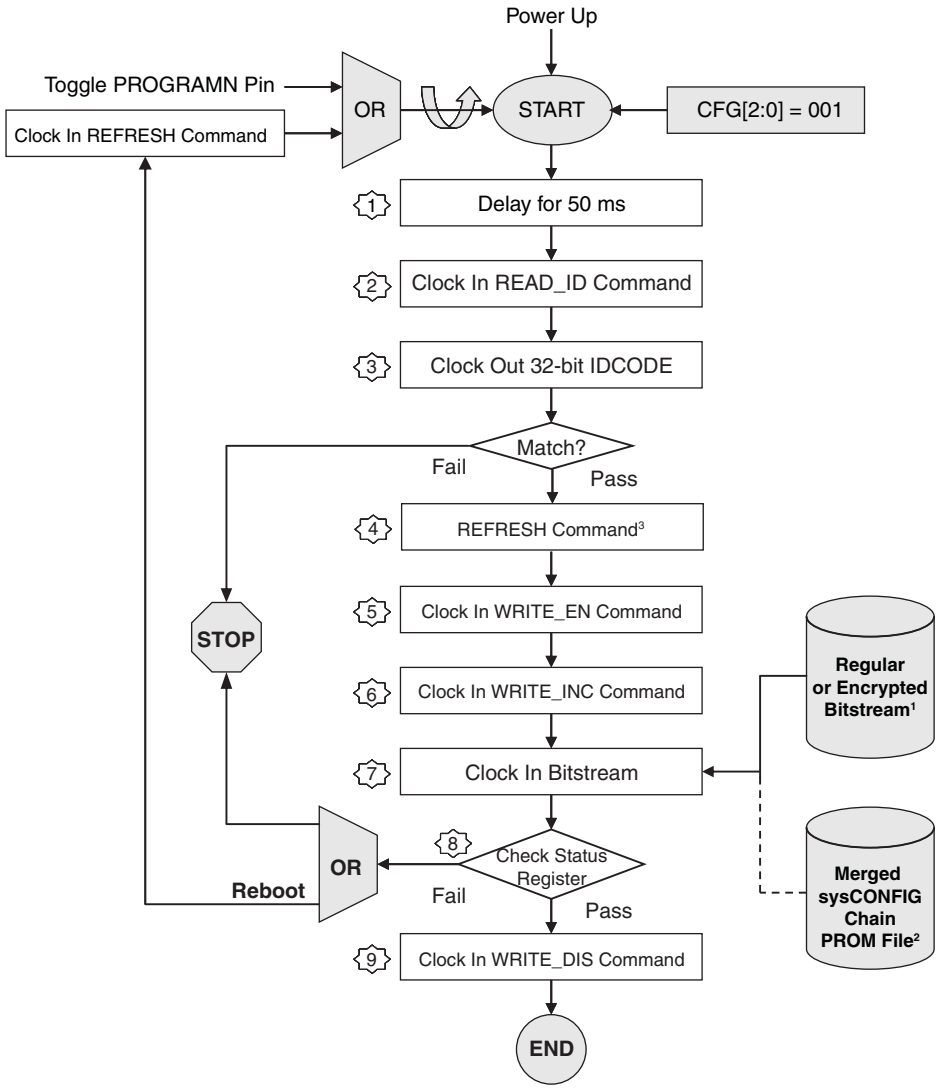
The Slave SPI port supports the regular LatticeECP3 bitstream and the encrypted bitstream (SSPI Mode).

The regular bitstream format is the same for all configuration modes. However, the encrypted bitstream format is configuration mode dependent. When generating an encrypted bitstream, the user must select the Slave SPI configuration mode (SSPI mode) for use with the Slave SPI port. The Encryption Key must also be pre-programmed into the device using JTAG mode.

The Slave SPI configuration flow diagram is shown in Figure 7. Highlights are listed below.

- The bitstream file is a stand-alone file. It is not part of the driver or system code. This provides seamless system integration and flexible file management. For example, the bitstream can be switched on the fly without changing a single line of system code.
- The LatticeECP3 device will wake up and enter User Mode once it reads in the entire bitstream. If it is necessary to delay the wake-up, the simplest method is by using the DONE pin. Wake-up can be delayed by holding the open-drain DONE pin low until the wake-up is desired.

Figure 7. Slave SPI Configuration Flow Diagram



Notes:

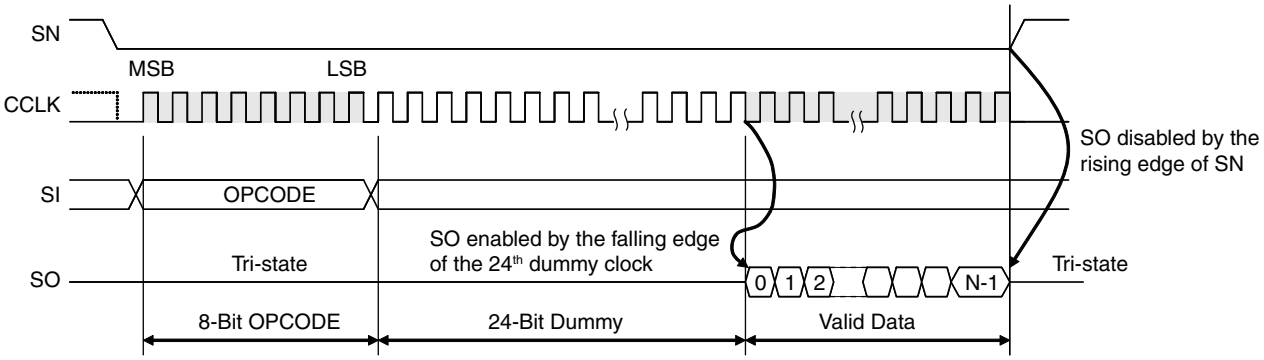
1. For a single LatticeECP3 device, the input file is a bitstream, which may be a standard or encrypted bitstream.
2. For a sysCONFIG chain of devices, the input file can be a merged PROM file. Refer to the “Advanced Applications – Slave SPI sysCONFIG Daisy Chaining” on page 21 for more details.
3. Use REFRESH command with No Delay as a Class C command instead of a Class D command. This REFRESH command is not for clearing all the SRAM, therefore no delay is needed.

Command Waveforms

Class A Command Waveforms

The Class A commands are ones that read data out from the ECP3 devices. Bit 0 of the data or bitstream will be read out first. The twenty four (24) dummy clocks provide the device the necessary delay for the proper execution of the command.

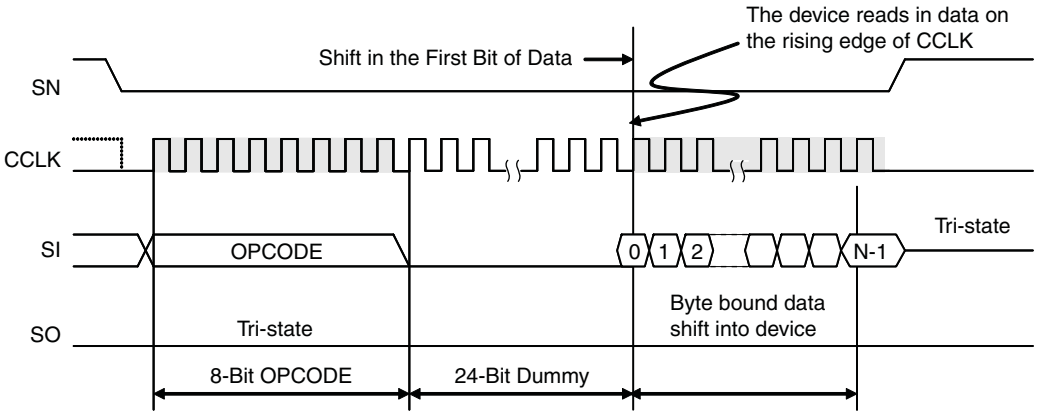
Figure 8. Class A Command Waveforms



Class B Command Waveforms

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

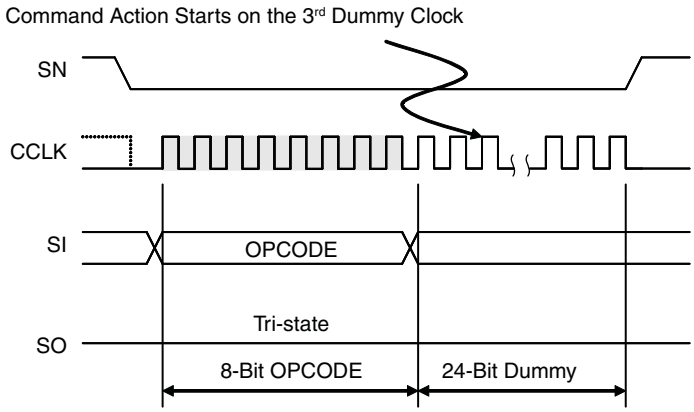
Figure 9. Class B Command Waveforms



Class C Command Waveforms

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

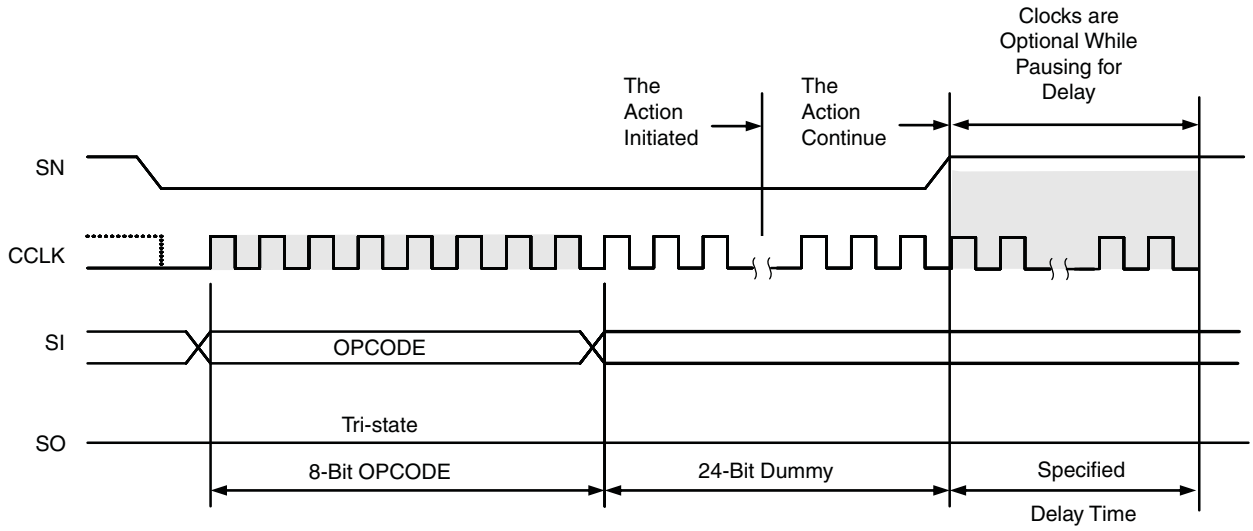
Figure 10. Class C Command Waveforms



Class D Commands Waveform

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

Figure 11. Class D Command Waveforms

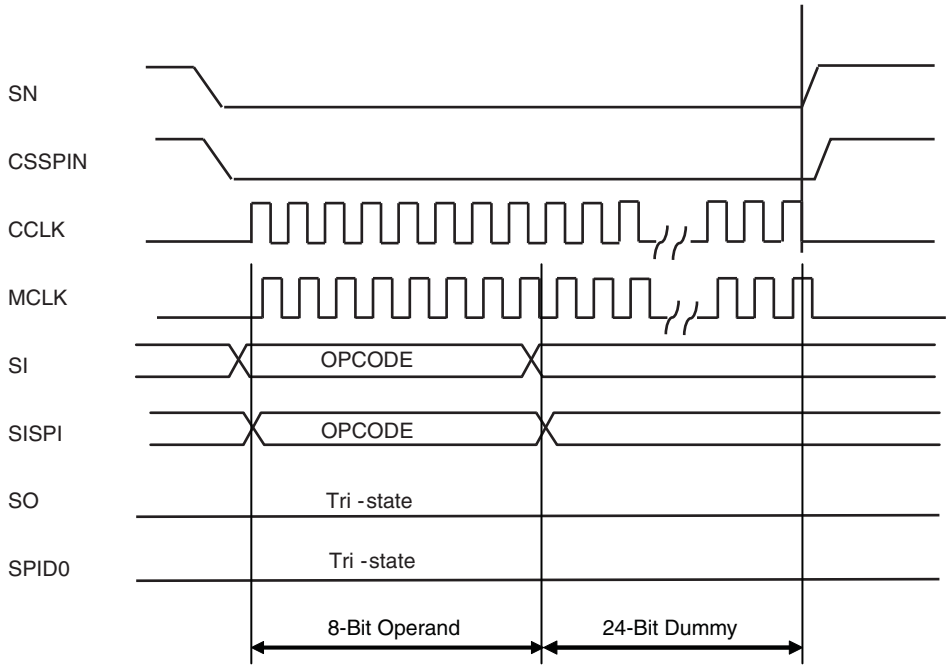


Class E Command Waveforms (Detailed Description of the PROGRAM_SPI Command)

The Class E commands are Lattice's specific commands that do not comply with the SPI industry standard. The PROGRAM_SPI command falls in this category. The purpose of the PROGRAM_SPI command is to allow the Slave SPI Port pins to be connected (effectively shorted) to their corresponding Master SPI port pins as follows:

- SI → SISPI
- SO ← SPID0
- SN → CSSPIN
- CCLK → MCLK

Figure 12. Class E Command Waveforms



The LatticeECP3 device will recognize this command only if the following conditions are met:

- The LatticeECP3 device configuration mode is SPI or SPIm.
- The LatticeECP3 device is erased, or the device is configured and SSPI Persistent is enabled in the bitstream.

Once the LatticeECP3 executes the command and the connection is made, the Slave SPI port can directly access the external SPI device to program the bitstream into the SPI Flash device.

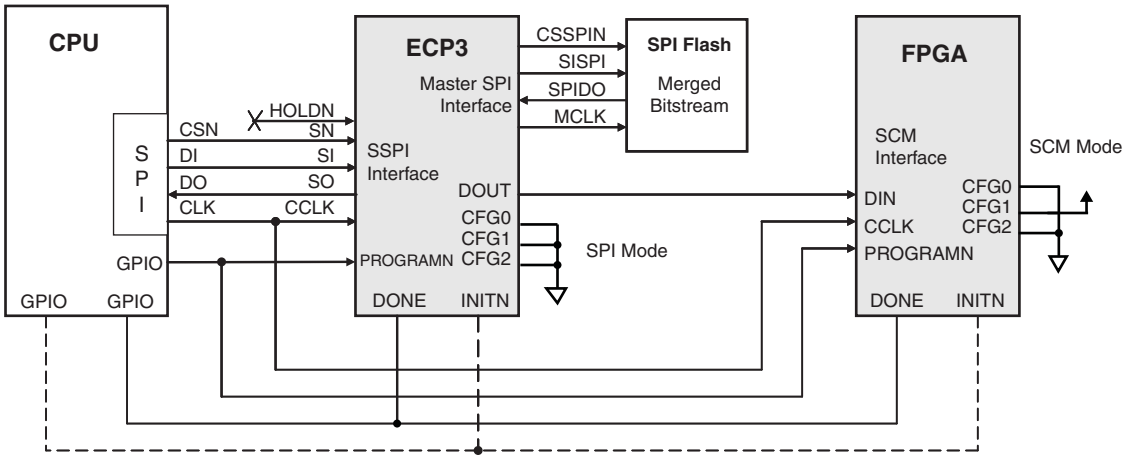
It is important to note that the Slave SPI port is effectively shorted to the Master SPI port. It means that the Slave SPI port of the LatticeECP3 device no longer pays attention to the commands and data coming in the port. They are sent right out to the SPI Flash device. In other words, the LatticeECP3 device will be in bypass mode after executing the PROGRAM_SPI command.

The only method to break the LatticeECP3 device from this bypass mode is by toggling the PROGRAMN pin or power cycling the device.

Since the LatticeECP3 device is set to SPI or SPIm mode, a standard SPI daisy chain can be formed, as shown in Figure 13. The standard SPI Flash daisy chain consists of a LatticeECP3 and another FPGA device in the Slave Serial Configuration (SCM) mode. The bitstream files of the daisy chained devices can be merged first using the ispUFW and then programmed into the SPI Flash device. Refer to TN1169, LatticeECP3 sysCONFIG Usage Guide

for more details. The PROGRAM_SPI command supports programming the SPI Flash device with either a single LatticeECP3, or a merged bitstream.

Figure 13. Standard SPI Daisy Chain



LatticeECP3 Bitstream File

Configuration Bitstream Format

The base binary file format is the same for all non-encrypted, non-1532 configuration modes. Different file types (hex, binary, ASCII, etc.) may ultimately be used to configure the device, but the data in the file is the same. Table 5 shows the format of a non-encrypted bitstream. The bitstream consists of a comment string, a header, the preamble, and the configuration setup and data.

Table 5. LatticeECP3 Bitstream File

Frame	Contents	Description
Comments	(Comment String)	ASCII Comment (Argument) String and Terminator
Header	1111...1111	24 Dummy bits
Preamble	1011110110110011	16-bit Standard Bitstream Preamble (0xBDB3)
Verify ID		64 bits of command and data
Store Compress		136 bits of command and data
Control Register 0		64 bits of command and data
NOOP		8 Dummy bits
Reset Address		32 bits of command and data
Write Increment		32 bits of command and data
Data 0		Data (*x* bits), 16-bit CRC, and Stop bits
Data 1		Data (*x* bits), 16-bit CRC, and Stop bits
. . .	. . .	. . .
Data *n*-1		Data (*x* bits), 16-bit CRC, and Stop bits
End	1111...1111	Terminator bits and 16-bit CRC
Usercode		64 bits of command and data
SED CRC		64 bits of command and data
Program Security		32 bits of command and data
EBR Write 0[2]		32 bits of command, EBR Initialization Data, 16-bit CRC, 32-bit Stop bits, 16-bit CRC
EBR Write 1[2]		32 bits of command, EBR Initialization Data, 16-bit CRC, 32-bit Stop bits, 16-bit CRC
. . .	. . .	. . .
EBR Write m-1[2]		32 bits of command, EBR Initialization Data, 16-bit CRC, 32-bit Stop bits, 16-bit CRC
Program Done		32 bits of command and data, 16-bit CRC
End	1111...1111	32-bit Terminator (all ones)

Notes:
1. The data in this table is intended for reference only.
2. The optional Write EBR frame is included for each initialized EBR.
3. Not all frames in the above File are required.

Read Back Bitstream Format

After issuing a READ_INC command, the configuration data can be shifted out of the Slave SPI port frame by frame. Table 6 shows the order that the configuration data is shifted out of the Slave SPI port. Each frame of data is preceded by one dummy byte of data.

Table 6. LatticeECP3 Slave SPI Read Back Data

Frame	Contents	Description
Dummy	XXXXXXXX	8 Dummy bits
Data 0		Configuration Data (*x* bits)
Dummy	XXXXXXXX	8 Dummy bits
Data 1		Configuration Data (*x* bits)
. . .	. . .	. . .
Dummy	XXXXXXXX	8 Dummy bits
Data *n*-1		Configuration Data (*x* bits)

The device specifics for the LatticeECP3 family are listed in Table 7.

Table 7. LatticeECP3 Device Specifics

Device	32-bit JTAG IDCODE	Configuration Frames (*n*)	Byte Bound Padding Bits	Data Bits per Frame (*w*)	Total Data Bits per Frame (*x*)	Standard Bitstream File Size	
						Configuration Data Only	Configuration + All EBR Initialization
ECP3-17	0x01010043	1543	0	2584	2584	4061960	4617800
ECP3-35	0x01012043	2067	4	3412	3416	7160872	8494888
ECP3-70	0x01014043	2819	4	6724	6728	19102328	23549048
ECP3-95	0x01014043	2819	4	6724	6728	19102328	23549048
ECP3-150	0x01015043	3607	4	8380	8384	30415008	37307424

Notes:

1. The EBR Initialization bitstream size depends on the number of EBR block(s) utilized. The size listed is the maximum size (all EBR block initialized).
2. The size of an encrypted bitstream file is approximately 50% larger than a standard bitstream file.
3. Since all data shifting operations must be byte bound, some devices require Padding Bits prior to shifting in the first real configuration data bits.

Advanced Applications – The Slave SPI Port and SPI Flash Interface

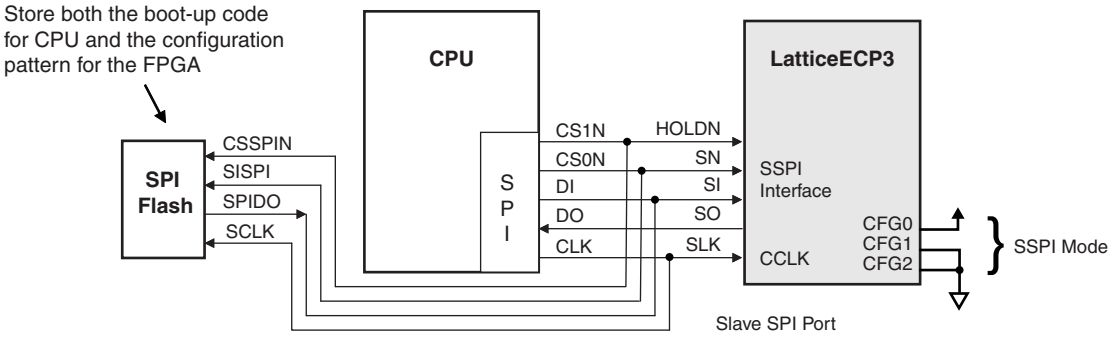
The above documentation provides Slave SPI port information for configuring a LatticeECP3 device.

The system diagram shown in Figure 14 illustrates an application of the Slave SPI interface. In this example, the CPU can independently access the external SPI flash for CPU boot-up code or for the FPGA bitstream. The FPGA selects Slave SPI as the primary boot source. When the CPU needs to configure the FPGA, it can drive CS0N low then shift the WRITE_EN into the device command to set the FGPA device into the Slave SPI configuration mode.

In this example, the CPU reads the configuration data from the SPI Flash and shifts it into the FPGA. There are two methods this can be done.

1. The CPU reads the entire bitstream from the SPI Flash before shifting it into the FPGA. This method is much simpler to support, but requires sufficient RAM to store the entire bitstream.
2. The CPU can read a frame of the bitstream from the SPI Flash device, shift the frame into the FPGA, and repeat for each frame. The method requires less RAM, and it does not require the CPU to have knowledge of the bitstream format. However it does require use of the HOLDN pin.

Figure 14. Example Slave SPI System Diagram (Slave SPI Mode)



Advanced Applications – Slave SPI Port and Master SPI Port

The system diagrams shown in Figure 15 and Figure 16 illustrate a primary application of the Slave SPI interface where the FPGA selects SPI Flash as the primary boot source. The CPU has the capability to program the SPI Flash device as well as command the FPGA to reboot from the SPI Flash by toggling the PROGRAMN pin. This requirement can only be met if the CPU drives the CCLK of the Slave SPI port, and MCLK drives the master clock out of the FPGA for the external SPI Flash device.

The bitstream programmed into the SPI Flash must set the "Persistent" to "SSPI". When the SPI Flash is blank, the LatticeECP3 device will be in the un-configured state, which has the same effect as when Persistent set to SSPI. Thus, there is no problem when programming the blank SPI Flash the first time. However, if field upgrade of the SPI Flash is required, then the "Persistent" must be set to "SSPI" in the bitstream. Otherwise, the Slave SPI port will be disabled and will not be available.

This configuration supports Dual Boot (SPIm mode), as shown in Figure 16. Refer to TN1216, LatticeECP3 and LatticeECP2/M Dual Boot Feature for more details.

This configuration also supports the formation of the standard SPI Flash daisy chain with LatticeECP3 or other FPGA devices in the Slave Serial Configuration (SCM) mode. The bitstream files of the daisy chained devices can be merged first using the ispUFW and then programmed into the SPI Flash device. Refer to TN1169, LatticeECP3 sysCONFIG Usage Guide for more details.

Figure 15. Example Slave SPI System Diagram (SPI Mode)

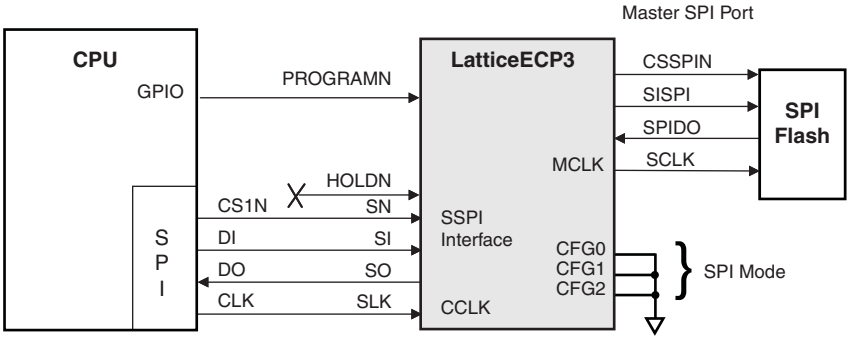
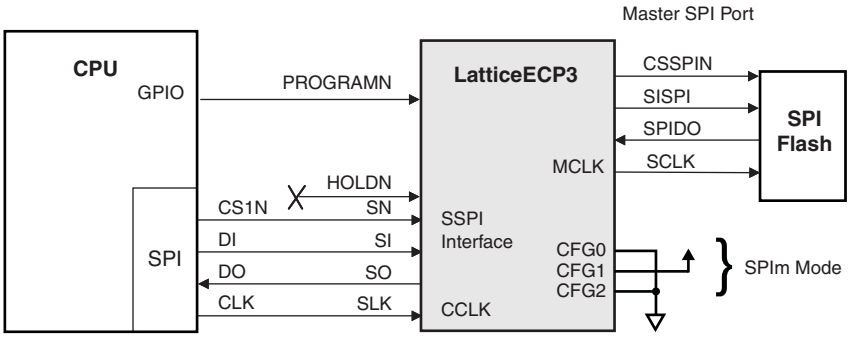


Figure 16. Example Slave SPI System Diagram (SPIm Mode)

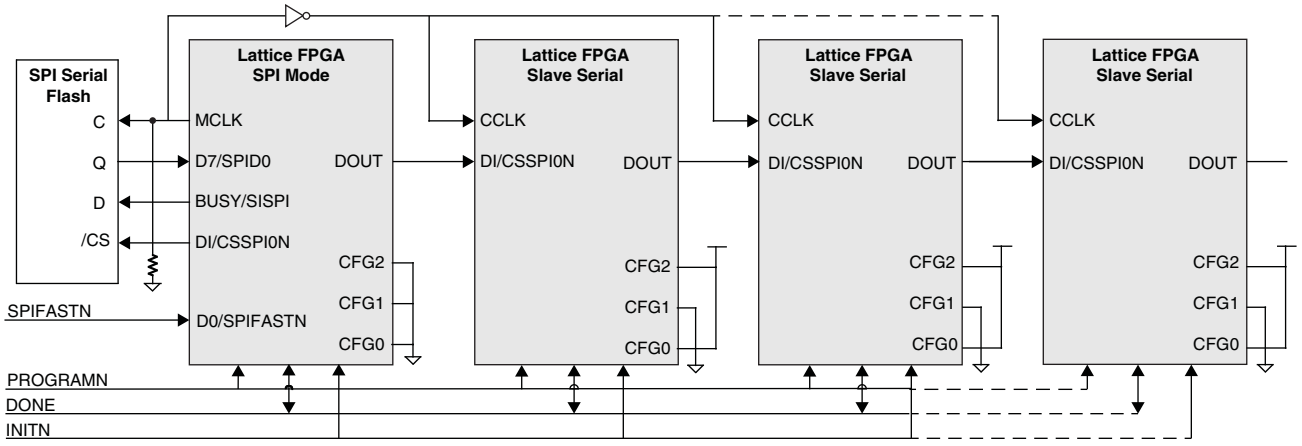


Advanced Applications – Master SPI sysCONFIG Daisy Chaining

With a sufficiently large SPI Flash, multiple FPGAs can be configured as shown in Figure 17. The requirements are listed below.

- The first device must be a LatticeECP3 device in Master SPI mode.
- The rest of the FPGA devices must be in Slave Serial Configuration (SCM) mode. The bitstream files for the daisy chain can be merged using the Deployment Tool.
- The DONE pin of all the FPGA devices must be connected together. This allows the devices to detect when the last device is done configuring.
- The 'Synchronous to External DONE Pin' option (DONE_EX) must be enabled in the Lattice Diamond™ Spreadsheet view, or the ispLEVER® Design Planner 'Global' tab, for all Lattice FPGA devices in the sysCONFIG chain.
- The Bypass option must be set by using the Diamond or ispLEVER 'Bitgen' properties for all Lattice FPGA devices in the sysCONFIG chain.

Figure 17. Master SPI sysCONFIG Daisy Chain



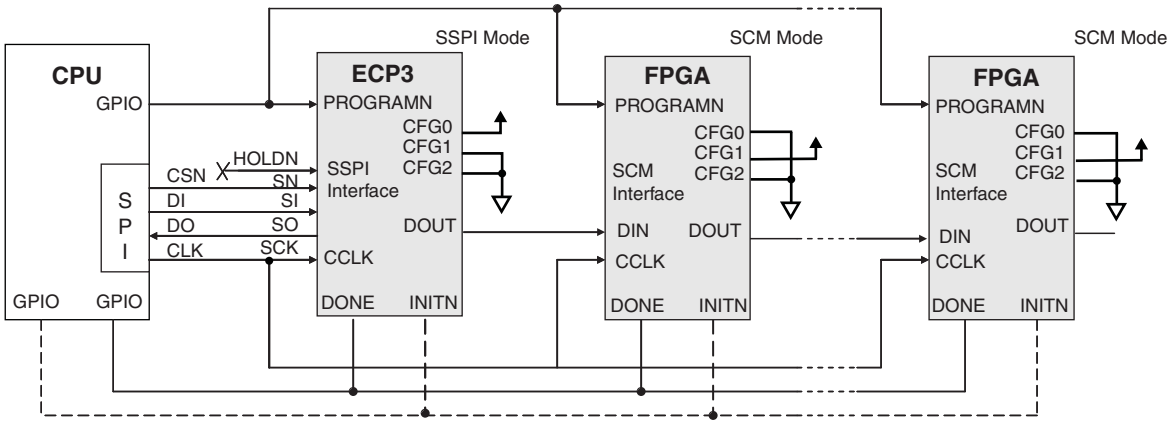
Note: The dashed lines indicate the connection is optional. The inverter for MCLK guarantees the hold time for data input to the daisy chained FPGAs.

Advanced Applications – Slave SPI sysCONFIG Daisy Chaining

Even though the Slave SPI port does not explicitly support daisy chaining, a sysCONFIG chain of other FPGA devices can be configured using the SPI port of a CPU. A block diagram is shown in Figure 18. The requirements are listed below.

- The first device must be a LatticeECP3 device in Slave SPI mode.
- The rest of the FPGA devices must be in Slave Serial Configuration (SCM) mode. The bitstream files for the daisy chain can be merged using the Deployment Tool.
- The DONE pin of all the FPGA devices must be connected together. This allows the devices to detect when the last device is done configuring.
- The 'Synchronous to External DONE Pin' option (DONE_EX) must be enabled in the Lattice Diamond Spreadsheet view, or the ispLEVER® Design Planner 'Global' tab, for all Lattice FPGA devices in the sysCONFIG chain.
- The Bypass option must be set by using the Diamond or ispLEVER 'Bitgen' properties for all Lattice FPGA devices in the sysCONFIG chain.

Figure 18. Slave SPI sysCONFIG Daisy Chain



Note: The dashed lines indicate the connection is optional.

Technical Support Assistance

For assistance, submit a technical support case at www.latticesemi.com/techsupport.

Revision History

Date	Version	Change Summary
August 2019	1.8	Changed document title to LatticeECP3 Slave SPI Port User Guide.
		Changed document number from TN1222 to FPGA-TN-02136.
		Updated the Method to Enable the Slave SPI Port section. Revised description of enabling Slave SPI persistence.
		Updated Technical Support Assistance information.
		Added Disclaimers section.
April 2014	01.7	Updated Table 7, LatticeECP3 Device Specifics. Changed ECP3-17 32-bit JTAG IDCODE.
		Updated Technical Support Assistance information.
June 2013	01.6	Updated Slave SPI Command Table and added note on operands.
		Updated Class E Command Waveforms diagram.
May 2013	01.5	Updated the Master SPI sysCONFIG Daisy Chain diagram and footnote.
	01.4	Added the Advanced Applications – Master SPI sysCONFIG Daisy Chaining section.
		Updated the Advanced Applications – Slave SPI sysCONFIG Daisy Chaining section.
		Updated the Slave SPI sysCONFIG Daisy Chain diagram.
April 2013	01.3	Updated Slave SPI Command Table.
		Added Slave SPI Command Usage Table.
		Updated Slave SPI Configuration Flow Diagram.
		Added Class E Commands Waveforms diagram.
		Updated Advanced Applications – Slave SPI Port and Master SPI Port section.
June 2012	01.2	Updated document with new corporate logo.
		Updated Slave SPI Read Waveforms diagram.
		Updated Slave SPI Write Waveforms diagram and added note afterward.
		Corrected column headings in AC Timing Requirements table (Min. and Max.).
November 2010	01.1	Removed EBR_READ command from the Slave SPI Command Table.
	01.0	Initial release.

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