Generic Soft SPI Master Controller Demo

User Guide

FPGA-UG-02123-1.0

December 2020

Disclaimers

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

Contents

Figures

Tables

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
FPGA	Field-Programmable Gate Array
LED	Light-emitting Diode
SCLK	Serial Clock Signal
SPI	Serial Peripheral Interface
SS	Slave Select
USB	Universal Serial Bus

1. Introduction

This document describes the setup procedures for the Generic Soft SPI Master Controller reference design demo using a MachXO3™-9400 Development Board to demonstrate simple transactions to an external SPI Flash device.

Figure 1.1 shows the MachXO3-9400 development board (LCMXO3LF-9400C-ASC-B-EVN) with an on-board external SPI Flash (Cypress S25L116K). This demo is not limited to this evaluation board and can be programmed into a wide array of Lattice FPGAs such as MachXO2™, MachXO3, LatticeECP3™, ECP5™, CrossLink™, CrossLink™-NX, and iCE40 UltraPlus™. Separate design projects are included to make it easier for you to custom fit this demo into their desired evaluation board.

For more information on the specific board used in this demo, visit:
http://www.latticesemi.com/products/developmentboardsandkits/machxo39400devboard.

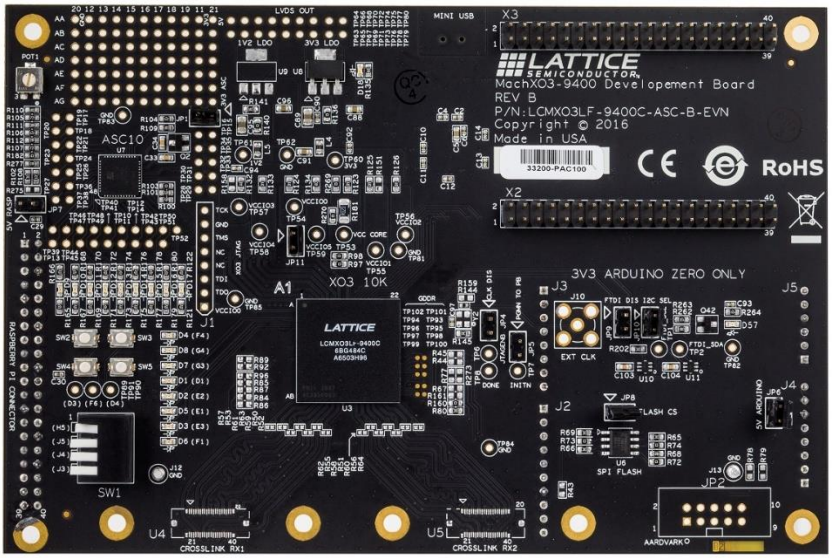


Figure 1.1. MachXO3-9400 Development Board

2. Functional Description

Figure 2.1 shows a simplified block diagram of this demo design. The Serial Peripheral Interface (SPI) bus provides an industry standard interface between processors and other devices and a good choice for designs that require full-duplex capability for sending and receiving data at the same time.

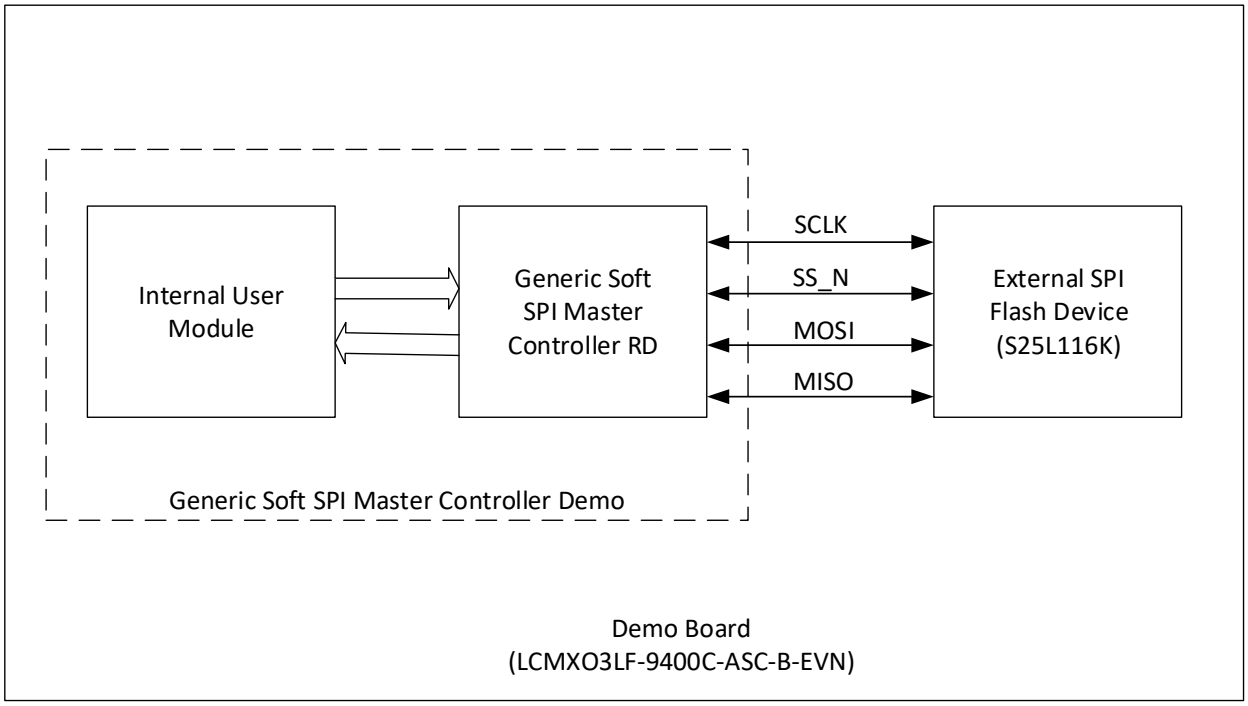


Figure 2.1. SPI Master Controller Demo Block Diagram

This demo implements the Generic Soft SPI Master Controller Reference Design(FPGA-RD-02209) by performing simple transactions to the external SPI Flash device found in the MachXO3-9400 Development Board and can be initiated using the onboard switches. These transactions are device specific, which means that if you are going to use another SPI Slave device, you should look into this slave device's datasheet and follow the proper command and data bytes framing.

You can control the switches located on the lower left portion of the demo board shown in Figure 2.2. To control this demo, you need to set the *ADR_OPT* switch to tell the SPI Master which *SS_N[4:0]* pin is going to be utilized (see Table 6.1). The type of transaction can be set using the *Transaction_Type[2:0]* switches (see Table 6.2). You need to hold the *START_N* switch button to start a SPI transaction with these selected settings. The Internal User module (*spi_master_controller_demo.v)* is utilizing a preloaded 512-byte EBR (Embedded Block RAM), which the SPI Master Demo design can fetch data from and eventually be sent to the *MOSI* pin during a write command.

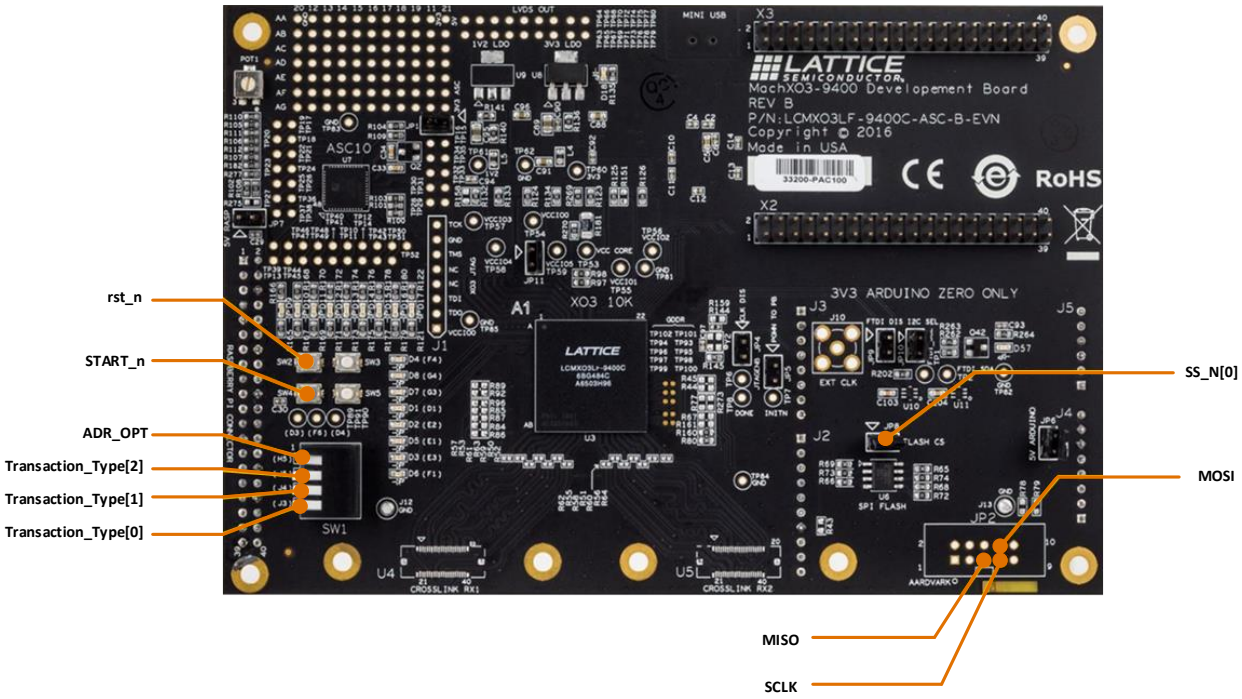


Figure 2.2. MachXO3-9400 Development Board Implementation

3. Demo Procedure

The following equipment is required for the demo:

- MachXO3-9400 Development Board (LCMXO3LF-9400C-ASC-B-EVN)
- Jumper wires
- Laptop/PC
- Logic Analyzer
- JED programming file
- USB 2.0 Type A to Mini-B cable
- Lattice Diamond® Programmer version 3.11 or higher

3.1. Demo Directory Structure

The demo design folder contains six subfolders: Bitstream, Docs, Project, Simulation, Source, and Testbench. The details of each subfolder are as follows:

- Bitstream – Contains the programming file for the MachXO3-9400 Development Board.
- Project – Contains subfolders for each FPGA Family. Each of these subfolders contains either a Diamond or a Radiant™ project file (.LDF and .RDF).
- Simulation – Contains subfolders for each FPGA Family. Each of these subfolders contains the simulation file (.DO) used to run RTL simulation on Aldec Active-HDL.
- Source – Contains all the Generic Soft SPI Master Demo RTL files.
- Testbench – Contains all the testbench files including the *SPI Slave (spi_slave.v).*

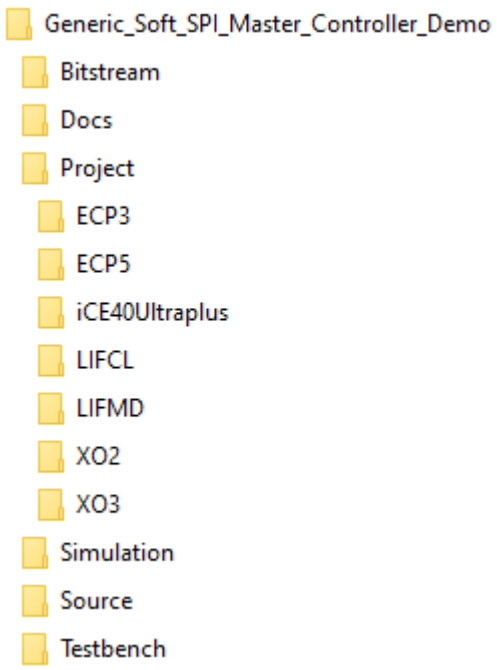


Figure 3.1. Packaged Design Directory Structure

4. Jumper Settings

Figure 4.1 shows the jumpers used in the demo board.

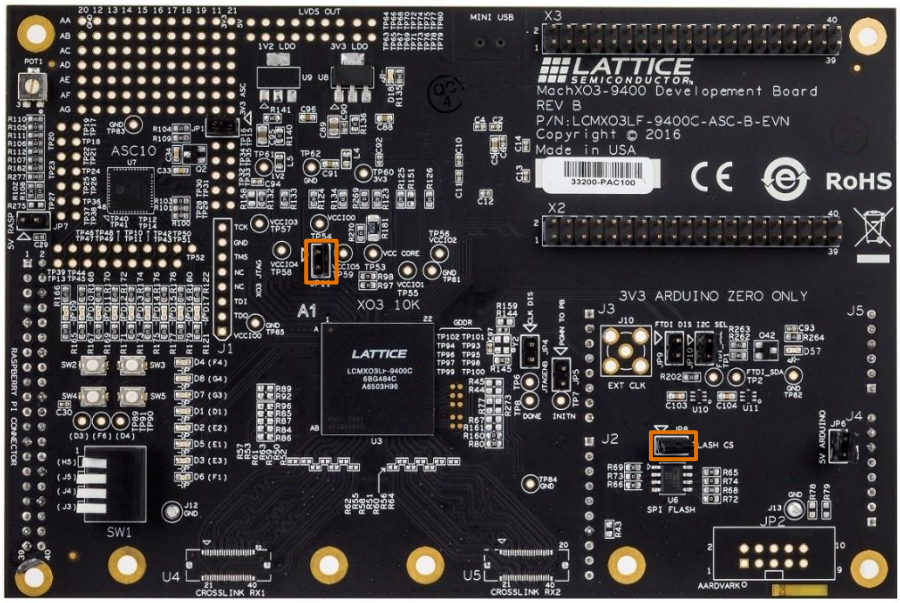


Figure 4.1. MachXO3-9400 Development Board Jumper Settings

Table 4.1. MachXO3-9400 Development Board Jumper Settings

Jumper	Description	Settings
JP11	12 MHz Oscillator	Default short (OSC connected) Alternate open (OSC unconnected) When the 12 MHz external oscillator is intended to be used, you need to uncomment the *EXT_CLK* compiler directive mentioned in Table 7.1.
JP8	SPI Flash SS_N Pin	Default short (SS_N[0] pin of the SPI Flash is connected to the FPGA).
—	—	All other jumpers should be kept open.

5. Programming the Board

To program the MachXO3-9400 development board:

1. Launch Diamond Programmer with **Create a new blank project**.

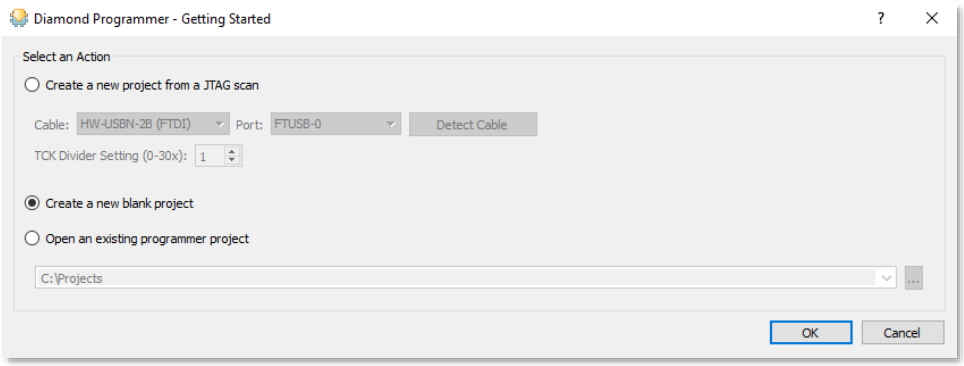


Figure 5.1. Create a New Blank Project

2. Select **MachXO3LF** for **Device Family** and **LCMXO3LF-9400C** for **Device**.

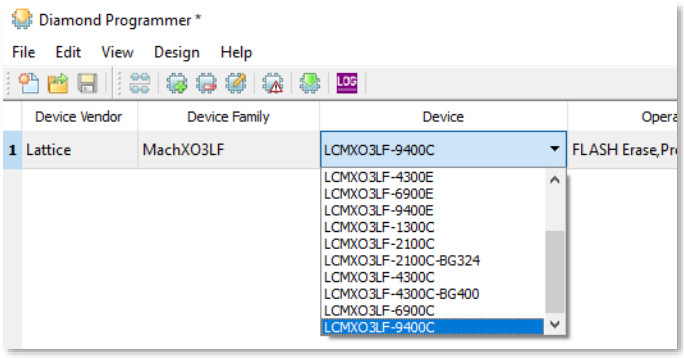


Figure 5.2. Device Selection

3. Right-click and select **Device Properties** and apply the settings as shown in Figure 5.3.

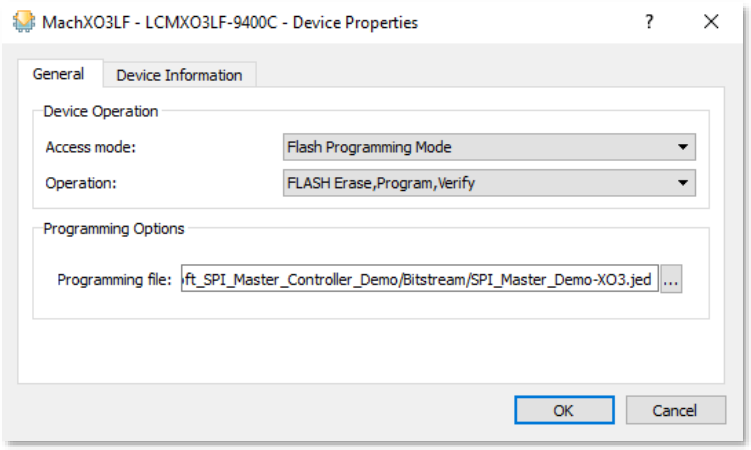


Figure 5.3. Device Properties

4. Click **OK** to close the Device Properties window.
5. Click the **Program** button  in the Diamond Programmer to start the Programming sequence.
6. When programming is successful, the output console displays the message shown on Figure 5.4. If programming fails, press **Detect Cable** in the right pane of the programmer as shown in Figure 5.5.

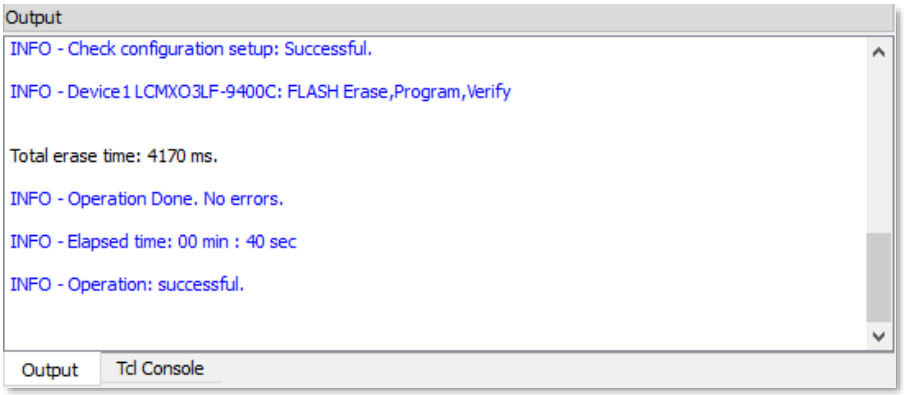


Figure 5.4. Output Console

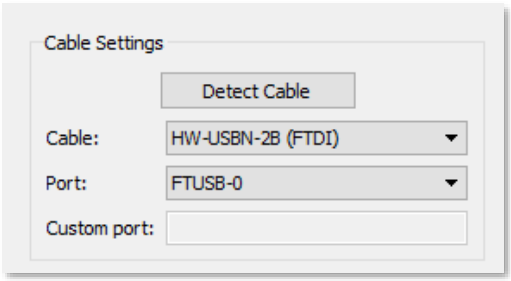


Figure 5.5. Detect Cable Button

6. Demo Procedure

To run the demo:

1. Connect the logic analyzer probes to the SPI signals (*SS_N[0], SCLK, MOSI, and MISO*). Ensure that the board and the logic analyzer has a common ground connection.

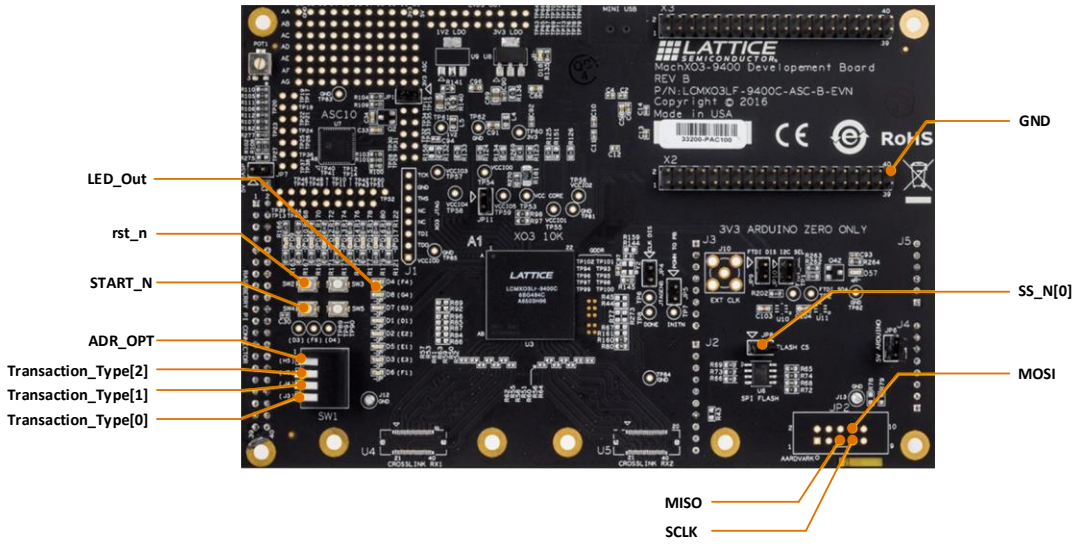


Figure 6.1. SS_N[0], SCLK, MOSI, MISO, and GND Connection Guide

2. Power up the board by using the USB Cable. D4 LED blinks if the board has been properly programmed.
3. Using Figure 6.2 and Table 6.1 as a guide, put ADR_OPT switch to 0 (down) so that the SPI Master uses *SS_N[0]* during a SPI transaction.
4. Using Figure 6.2 and Table 6.2 as a guide, select the desired transaction type and press START_N until a SPI transaction is registered in the logic analyser as shown in Figure 6.3, Figure 6.4, Figure 6.5, Figure 6.6, and Figure 6.7.

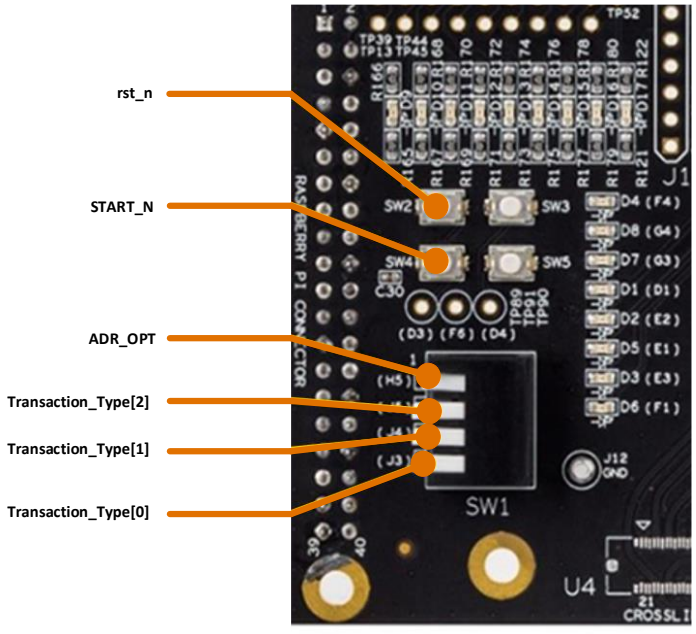


Figure 6.2. SPI Master Control Switches

Table 6.1. Slave Address Options

ADR_OPT	SS_N[4:0] Bit to be Used*
0	SS_N[0]
1	SS_N[1]

***Note:** SS_N[4:2] are not used in this demo.

Table 6.2. Slave Address Options

Transaction_Type[2]	Transaction_Type[1]	Transaction_Type[0]	Command Name	First Byte Instruction[1]
0	0	0	Device ID	90h
0	0	1	Write Enable	06h
0	1	0	Write Disable	04h
0	1	1	Write Data	02h
1	0	0	Read Data	03h
1	0	1	Block Erase[2]	D8h

Notes:

1. Byte instruction may vary across different slave devices. This demo design is using the instructions for Cypress S25L116K SPI Flash. Refer to the datasheet for more info.
2. Block Erase can only be performed after performing Write Enable command. After successfully performing the Block Erase command, the SPI flash automatically locks the device to prevent writing (Write Disable).

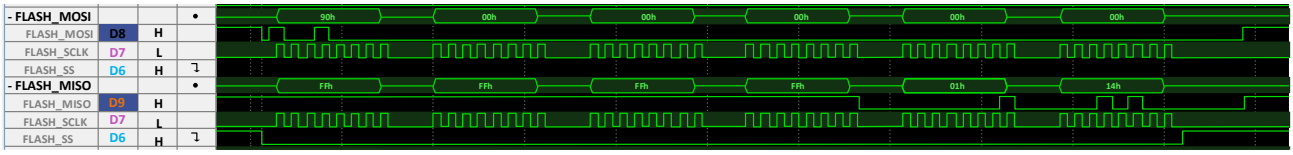


Figure 6.3. SPI Transaction Waveform During Device ID Command (S25L116K)

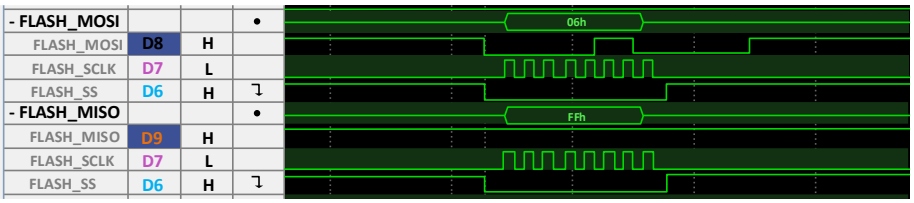


Figure 6.4. SPI Transaction Waveform During Write Enable Command (S25L116K)

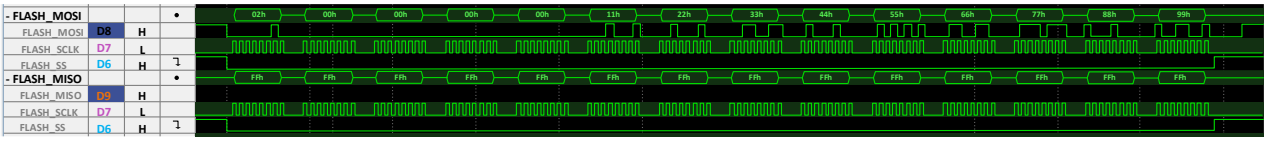


Figure 6.5. SPI Transaction Waveform During Write Data Command (S25L116K)

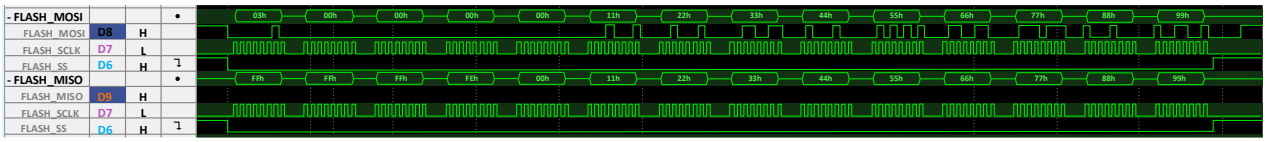


Figure 6.6. SPI Transaction Waveform During Read Data Command (S25L116K)

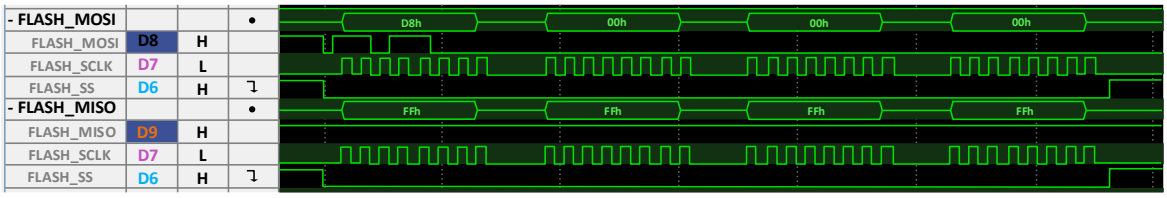


Figure 6.7. SPI Transaction Waveform During Block Erase Command (S25L116K)

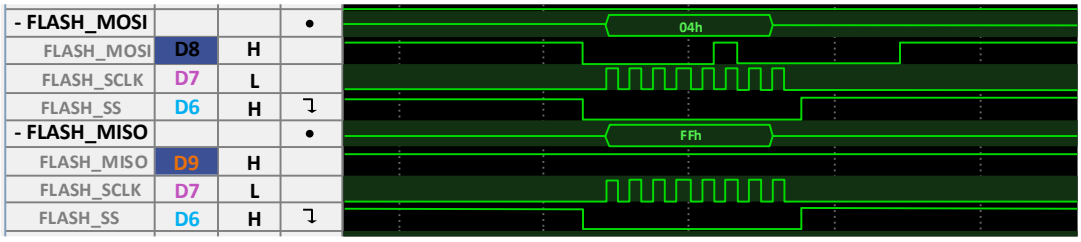


Figure 6.8. SPI Transaction Waveform During Write Disable Command (S25L116K)

7. Customization

To customize this demo design, a file named *tb_defines.v* in the testbench folder contains all the compiler directives that you can modify. This includes device selection, slave addresses settings, clock source, clock speed, and others. Table 7.1 shows the complete list of compiler directives. Figure 7.1 shows an example of customization implemented in the *tb_defines.v* file.

Table 7.1. Compiler Directives

Category	Compiler Directives	Remarks
Simulation Test	SIM_TEST	Uncomment to enable simulation. This disables the debounce logic and shorten the time needed before waveforms are displayed during simulation. When generating a bitstream, you should comment this out to enable the debounce logic and prevent spurious SPI transactions in the actual hardware.
Device Selection	ECP3	Uncomment only one to enable the selected device. The default for this demo is MachXO3.
	ECP5	
	LIFMD	
	LIFCL	
	MachXO2	
	MachXO3	
	iCE40 UltraPlus	
Slave Selection	SLAVE_A	Defines the input for the *i_SS_cfg* port and determines which bit of the *SS_N[4:0]* is activated during a SPI transaction. In this demo, the ADR_OPT switch determines whether SLAVE_A (*SS_N[0]*) or SLAVE_B (*SS_N[1]*) is used in the SPI transaction.
	SLAVE_B	
SPI Mode	SPI_MODE_0	Uncomment only one to enable the selected SPI Mode with defined CPOL and CPHA combinations. The default for this demo is SPI_MODE_0.
	SPI_MODE_1	
	SPI_MODE_2	
	SPI_MODE_3	
Clock Selection	EXT_CLK	Uncomment if external clock is desired. If internal clock is selected, only select 24 MHz in the Clock Speed selection. The default for this demo is using the internal oscillator (EXT_CLK is commented out).
Clock Speed Selection	CLK_12MHZ	Uncomment only one to enable the selected clock speed. If the desired clock speed is not in the selection, you need to modify the testbench. The default for this demo is 24 MHz.
	CLK_24MHZ	
	CLK_32MHZ	
SCLK Frequency Selection	CLOCK_SEL	Generates the SCLK clock frequency in terms of system clock cycles using the formula: SCLK=clk/2*(CLOCK_SEL+1) Allowable value is from 0 to 255.
Byte Interval	BYTE_INTERVAL	Defines the interval between SPI data bytes in terms of system clock cycles. Allowable value is from 1 to 255.
SS to SCLK Interval	SS_SCLK_INTERVAL	Defines the interval between the SS_N assertion to zero and the first SCLK edge. Allowable value is from 1 to 255.
SCLK to SS Interval	SCLK_SS_INTERVAL	Defines the interval between the SS_N deassertion to high and the last SCLK edge. Allowable value is from 1 to 255.

```
1  // ***********************************************
2  // Simulation Test
3  // (Uncomment to enable simulation.)
4  // ***********************************************
5  //`define SIM_TEST
6
7
8  // ***********************************************
9  // Device Selection
10 // (Uncomment the selected device.)
11 // ***********************************************
12 //`define ECP3
13 //`define ECP5
14 //`define LIFMD
15 //`define LIFCL
16 //`define XO2
17 `define XO3
18 //`define Ultraplus
19
20
21 // ***********************************************
22 // Slave Selection
23 // (Defines Wh)
24 // ***********************************************
25 `define SLAVE_A     5'b00001
26 `define SLAVE_B     5'b00010
27
28
29 // ***********************************************
30 // SPI Mode
31 // (Enter the desired value.)
32 // ***********************************************
33 `define SPI_MODE_0          // CPOL == 0, CPHA == 0,
34 //`define SPI_MODE_1            // CPOL == 0, CPHA == 1,
35 //`define SPI_MODE_2            // CPOL == 1, CPHA == 0,
36 //`define SPI_MODE_3            // CPOL == 1, CPHA == 1,
37
38 // ***********************************************
39 // Data Direction
40 // (Enter the desired value.)
41 // ***********************************************
42 `define DIRECTION       1'b0    // 1'b0 = MSB First, 1'b1 = LSB First
43
44
45
46 // ***********************************************
47 // Clock Selection
48 // (Uncomment to enable external clock.)
49 // ***********************************************
50 //`define EXT_CLK // Only select 24MHz in the Clock Speed Selection when using the internal oscillator.
51
52
53 // ***********************************************
54 // Clock Speed Selection
55 // (Uncomment the selected clock speed.)
56 // ***********************************************
57 //`define CLK_12MHZ
58 `define CLK_24MHZ
59 //`define CLK_32MHZ
60
61 // ***********************************************
62 // SPI Clock and Tiiming Parameters
63 // (Enter the desired value).)
64 // ***********************************************
65 `define CLOCK_SEL           1   // From 0 to 255. SCLK=clk/2*(CLOCK_SEL+1)
66 `define BYTE_INTERVAL       1   // From 1 to 255. Defines the interval between SPI data bytes in terms of
67 `define SS_SCLK_INTERVAL    1   // From 1 to 255. Defines the interval between the SS_N assertion to low a
68 `define SCLK_SS_INTERVAL    1   // From 1 to 255. Defines the interval between the SS_N deassertion to hig
```

Figure 7.1. Compiler Directives Customization Example

Appendix A. Simulation Procedures Using the DO file

To use the simulation DO file, perform the following steps:

1. Before running the simulation, make sure *SIM_TEST* is not commented out in the *tb_defines.v* source as shown in Figure A.1.

```
1  // ************************************************
2  // Simulation Test
3  // (Uncomment to enable simulation.)
4  // ************************************************
5  `define SIM_TEST
```

Figure A.1. SIM_TEST Commented-Out in the *tb_defines.v* Source File

2. Open the DO file on a text editor and replace the text **<ENTER simulation DIRECTORY PATH HERE>** from Line 1 with the directory path of the simulation file. An example is seen on Line 4 of the file.

rtl_verilog.do

```
1  set SIM_DIR "<ENTER simulation DIRECTORY PATH HERE>"
2
3  # Example:
4  # set SIM_DIR "D:/Generic_Soft_SPI_Master_Controller_Demo/Simulation/XO3"
```

Figure A.2. Changing the Simulation Directory

3. Run the file on Aldec Active-HDL by selecting *Execute macro…* under the **Tools** option.

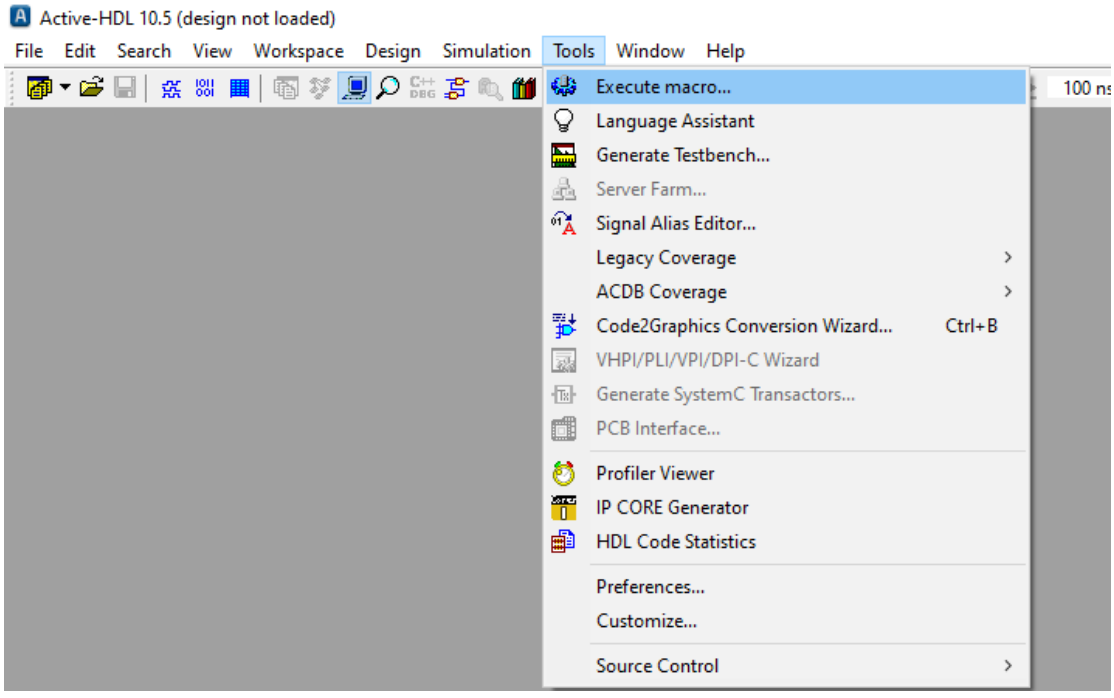
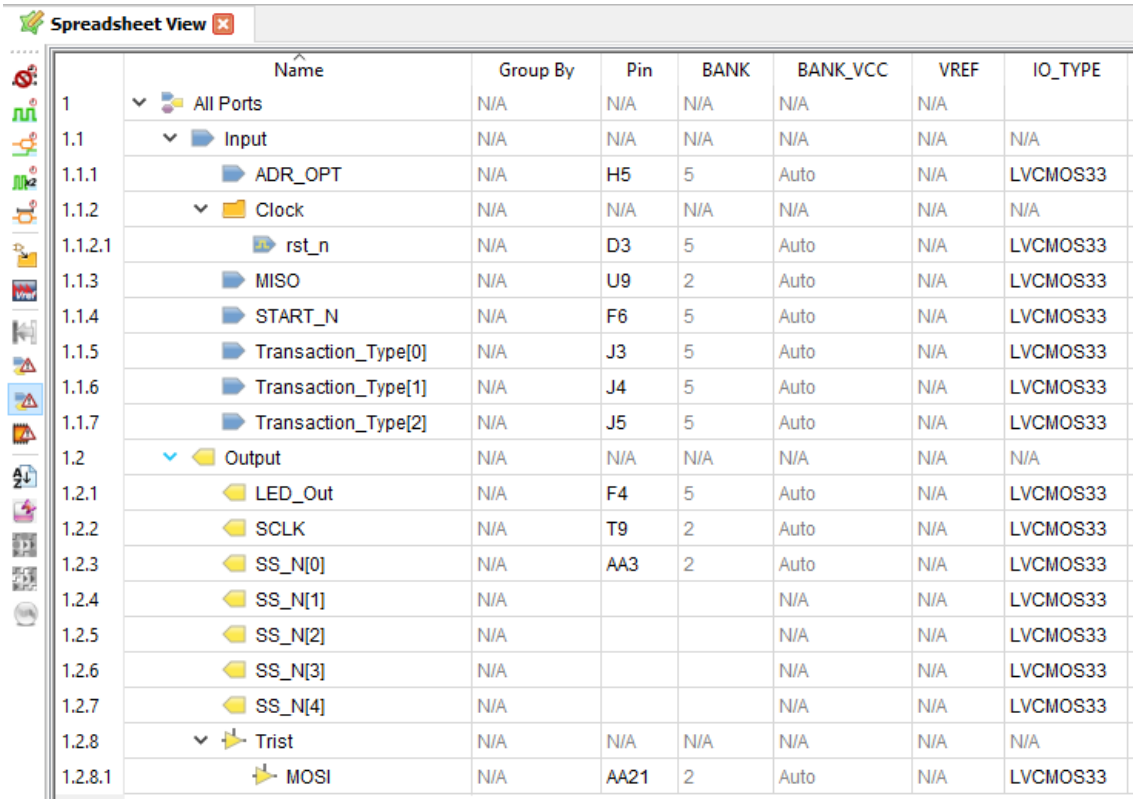


Figure A.3. Running the Simulation File

Appendix B. Port Assignments

Most of the projects included in this demo have no port assignments. You should manually assign them based on their selected device and package option. However, the included demo bitstream (SPI_Master_Demo-XO3.jed) is compiled using the port assignments and settings as shown in Figure B.1.



	Name	Group By	Pin	BANK	BANK_VCC	VREF	IO_TYPE
1	All Ports	N/A	N/A	N/A	N/A	N/A	
1.1	Input	N/A	N/A	N/A	N/A	N/A	N/A
1.1.1	ADR_OPT	N/A	H5	5	Auto	N/A	LVCMOS33
1.1.2	Clock	N/A	N/A	N/A	N/A	N/A	N/A
1.1.2.1	rst_n	N/A	D3	5	Auto	N/A	LVCMOS33
1.1.3	MISO	N/A	U9	2	Auto	N/A	LVCMOS33
1.1.4	START_N	N/A	F6	5	Auto	N/A	LVCMOS33
1.1.5	Transaction_Type[0]	N/A	J3	5	Auto	N/A	LVCMOS33
1.1.6	Transaction_Type[1]	N/A	J4	5	Auto	N/A	LVCMOS33
1.1.7	Transaction_Type[2]	N/A	J5	5	Auto	N/A	LVCMOS33
1.2	Output	N/A	N/A	N/A	N/A	N/A	N/A
1.2.1	LED_Out	N/A	F4	5	Auto	N/A	LVCMOS33
1.2.2	SCLK	N/A	T9	2	Auto	N/A	LVCMOS33
1.2.3	SS_N[0]	N/A	AA3	2	Auto	N/A	LVCMOS33
1.2.4	SS_N[1]	N/A			N/A	N/A	LVCMOS33
1.2.5	SS_N[2]	N/A			N/A	N/A	LVCMOS33
1.2.6	SS_N[3]	N/A			N/A	N/A	LVCMOS33
1.2.7	SS_N[4]	N/A			N/A	N/A	LVCMOS33
1.2.8	Trist	N/A	N/A	N/A	N/A	N/A	N/A
1.2.8.1	MOSI	N/A	AA21	2	Auto	N/A	LVCMOS33

Figure B.1. MachXO3 Port Assignment Using Diamond Spreadsheet View

References

For more information, refer to the following documents:

- LatticeECP3 EA Family Data Sheet (DS1021)
- ECP5 and ECP5-5G Family Data Sheet (FPGA-DS-02012)
- CrossLink Family Data Sheet (FPGA-DS-02007)
- CrossLink-NX Family Data Sheet (FPGA-DS-02049)
- MachXO2 Family Data Sheet (DS1035)
- MachXO3 Family Data Sheet (FPGA-DS-02032)
- iCE40 UltraPlus Family Data Sheet (FPGA-DS-02008)
- Generic Soft SPI Master Controller (FPGA-RD-02209)

Technical Support

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

Revision History

Revision 1.0, December 2020

Section	Change Summary
All	Initial release

www.latticesemi.com