



Soft I²C Master and Slave Simple Write-Read Demo

User Guide

FPGA-UG-02122-1.0

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

Acronyms in This Document	5
1. Introduction	6
2. Functional Description	7
2.1. I ² C Master Board	7
2.2. I ² C Slave Board	8
3. Demo Requirements	9
3.1. Demo Directory Structure	9
4. Jumper Settings	10
5. Programming the Boards	12
5.1. MachXO3-9400 Development Board (I ² C Master)	12
5.2. iCE40 UltraPlus Breakout Board (I ² C Slave)	14
6. Demo Procedure	16
7. Customization	18
Appendix A. Simulation Procedures Using the DO file	20
Appendix B. Port Assignments	21
References	22
Technical Support	23
Revision History	24

Figures

Figure 1.1. MachXO3-9400 Development Board	6
Figure 1.2. iCE40 UltraPlus Breakout Board	6
Figure 2.1. 4:1 Master and Slave Block Diagram	7
Figure 2.2. 4:1 MachXO3-9400 Development Board Implementation	7
Figure 2.3. iCE40 UltraPlus Breakout Board Implementation	8
Figure 3.1. Packaged Design Directory Structure	9
Figure 4.1. MachXO3-9400 Development Board Jumper Settings	10
Figure 4.2. iCE40 UltraPlus Breakout Board Jumper Settings	10
Figure 5.1. Create a New Blank Project	12
Figure 5.2. Device Selection	12
Figure 5.3. Device Properties	13
Figure 5.4. Output Console	13
Figure 5.5. Create a New Blank Project	14
Figure 5.6. Device Selection	14
Figure 5.7. Device Properties	15
Figure 5.8. Output Console	15
Figure 6.1. SCL and SDA Connections between the Master and the Slave	16
Figure 6.2. Slave Address and Transaction Settings	16
Figure 6.3. I ² C Write Transaction	17
Figure 6.4. I ² C Read Transaction	17
Figure 6.5. I ² C Write Transaction with Repeat Start	17
Figure 7.1. Compiler Directives Customization Example	19
Figure A.1. SIM_TEST Commented-Out in the <i>tb_defines.v</i> Source File	20
Figure A.2. Changing the Simulation Directory	20
Figure A.3. Running the Simulation File	20
Figure B.1. MachXO3 Port Assignment Using Diamond Spreadsheet View	21
Figure B.2. iCE40 UltraPlus Port Assignment Using Diamond Spreadsheet View	21

Tables

Table 4.1. MachXO3-9400 Development Board Jumper Settings	10
Table 4.2. iCE40 UltraPlus Breakout Board Jumper Settings	11
Table 6.1. Slave Address Options	17
Table 6.2. Slave Address Options	17
Table 7.1. Compiler Directives	18

Acronyms in This Document

A list of acronyms used in this document.

Acronym	Definition
FPGA	Field Programmable Gate Array
I ² C	Inter-Integrated Circuit
LED	Light-emitting diode
OSC	Oscillator
USB	Universal Serial Bus

1. Introduction

This document describes the setup procedures for the Generic Soft I²C Master Controller and Generic Soft I²C Slave Peripheral reference design demos using a MachXO3-9400 development board and an iCE40 UltraPlus™ breakout board to demonstrate simple write and read transactions.

Figure 1.1 and Figure 1.2 shows the MachXO3-9400 Development Board (LCMX03LF-9400C-ASC-B-EVN) and the iCE40 UltraPlus breakout board (iCE40UP5K-B-EVN). This demo is not limited to these evaluation boards 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 boards used in this demo, visit the following web pages:

- <http://www.latticesemi.com/products/developmentboardsandkits/machxo39400devboard>
- <http://www.latticesemi.com/products/developmentboardsandkits/ice40ultraplusbreakoutboard>

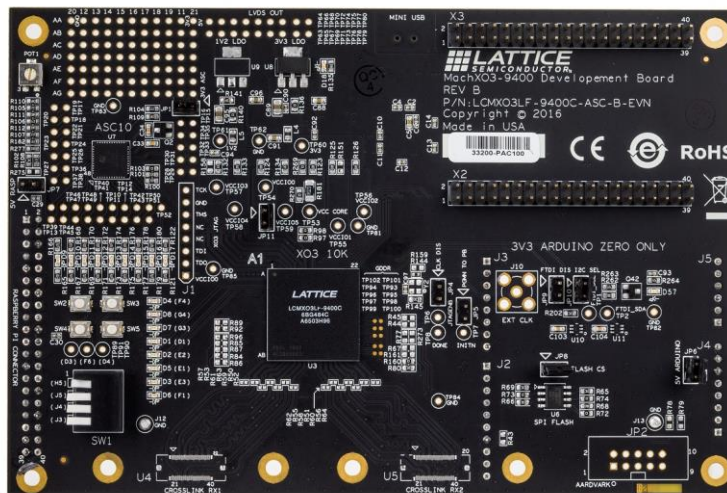


Figure 1.1. MachXO3-9400 Development Board

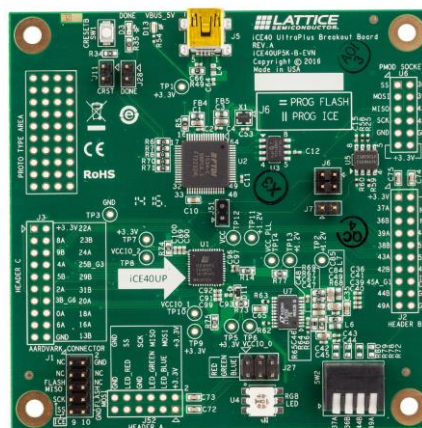


Figure 1.2. iCE40 UltraPlus Breakout Board

2. Functional Description

I²C or Inter-Integrated Circuit is a popular serial interface protocol that is widely used in many electronic systems. The I²C interface is a two-wire interface capable of half-duplex serial communication at moderate to high speeds of up to a few megabits per second. There are thousands of I²C peripherals on the market today, ranging from data converters to video processors. The I²C bus is a good choice for designs that need to communicate with low-speed peripherals due to its simplicity and low cost.

This demo integrates both Generic Soft I²C Master Controller and Generic Soft I²C Slave Peripheral reference designs by performing simple write and read transactions.

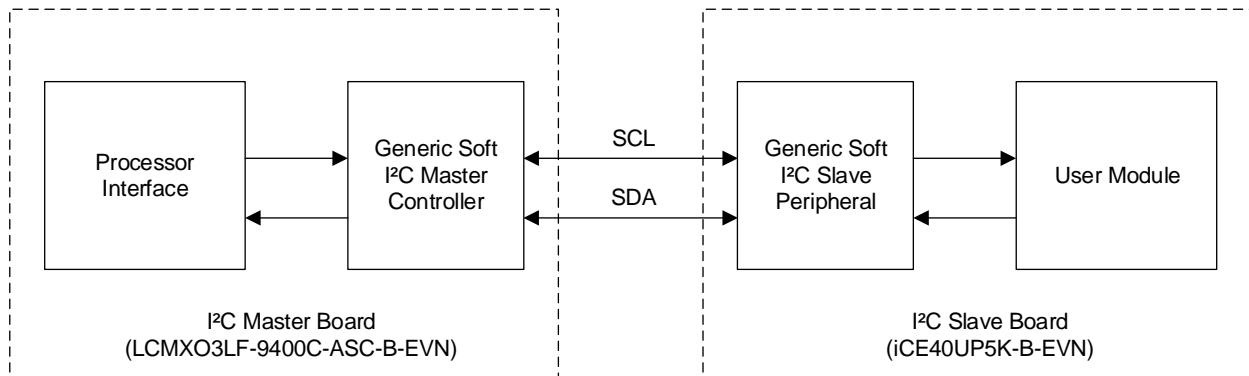


Figure 2.1. 4:1 Master and Slave Block Diagram

2.1. I²C Master Board

The I²C Master demo allows you to perform simple I²C transactions which can be initiated using the on-board switches of the MachXO3-9400 development board. You need to set the ADR_MODE and ADR_OPT switches to tell the I²C Master which I²C Slave Address is going to be accessed (see Table 6.1). The type of transaction can be set using the Transaction_Type[1:0] switches (see Table 6.2). You need to hold the BEGIN switch button to start an I²C transaction with the selected settings. The Processor Interface (*i2c_ebr_master_top*) is utilizing a preloaded 512-byte EBR (Embedded Block RAM), which the I²C Master design can fetch data from and eventually be sent to the I²C Bus during write transaction.

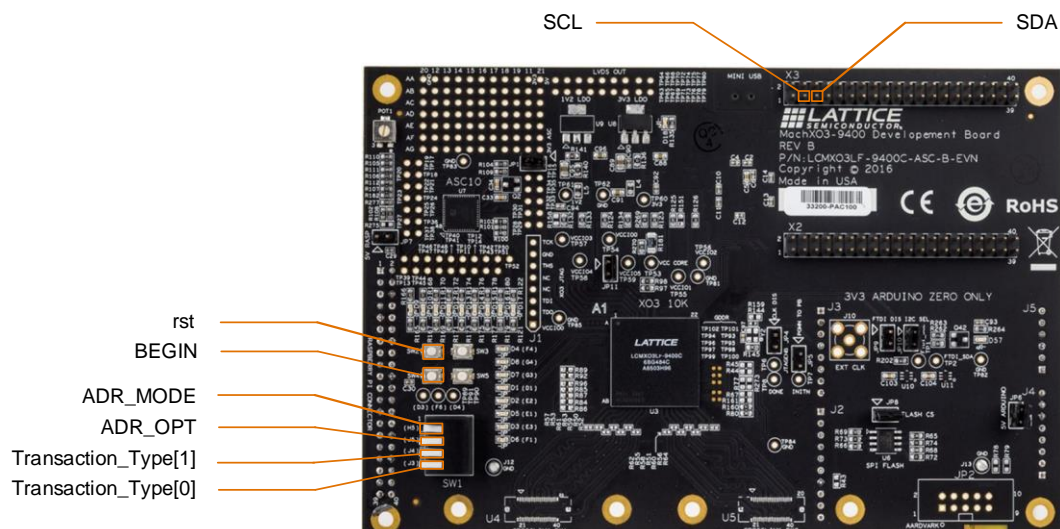


Figure 2.2. 4:1 MachXO3-9400 Development Board Implementation

2.2. I²C Slave Board

The I²C Slave demo is loaded to an iCE40 UltraPlus breakout board. The design also has a user module (*i2c_ebr_slave.v*) that utilizes a preloaded 512-byte EBR which the I²C master can either read directly, or write new data into. The I²C slave demo is designed to be very simple and only stores the data sent by the I²C Master. Whenever the I²C Master sends a read request, the I²C Slave sends back the previously stored data sent by the Master.

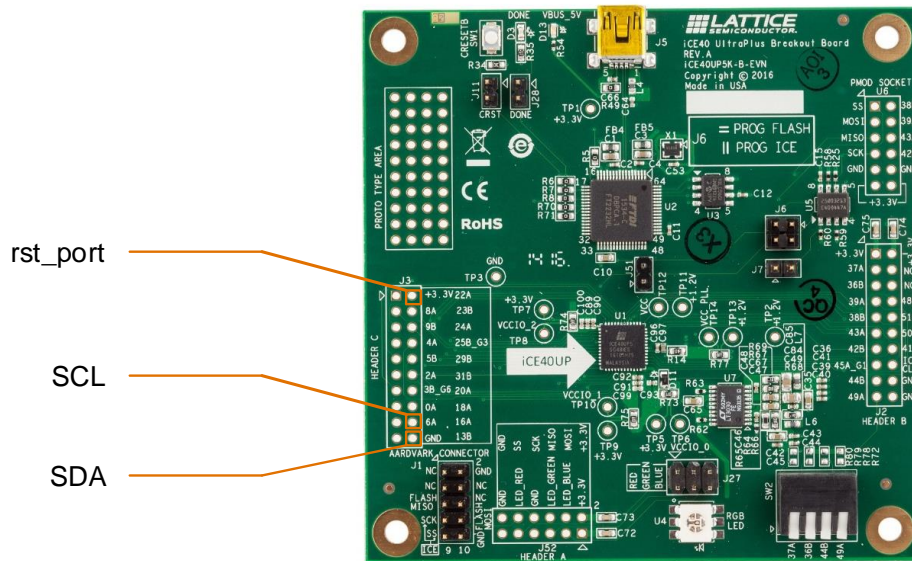


Figure 2.3. iCE40 UltraPlus Breakout Board Implementation

3. Demo Requirements

The following equipment is required for the demo:

- MachXO3-9400 Development Board (LCMXO3LF-9400C-ASC-B-EVN)
- iCE40 UltraPlus Breakout Board (iCE40UP5K-B-EVN)
- Jumper wires.
- Laptop/PC
- Logic Analyzer
- JED and BIN programming files.
- USB 2.0 Type A to Mini-B cable
- Lattice Radiant™ Programmer version 2.1 or higher
- 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 files for the MachXO3-9400 Development Board (Master) and iCE40 UltraPlus Breakout Board (Slave).
- 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 I²C Master and Slave RTL files.
- Testbench – contains all the testbench (*tb.v*) and the compiler directive (*tb_defines.v*) files.

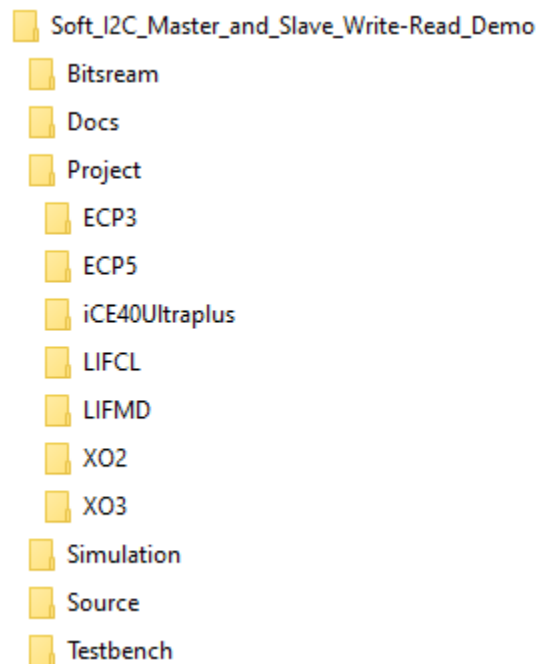


Figure 3.1. Packaged Design Directory Structure

4. Jumper Settings

Figure 4.1 and Figure 4.2 show the two boards used in the demo.

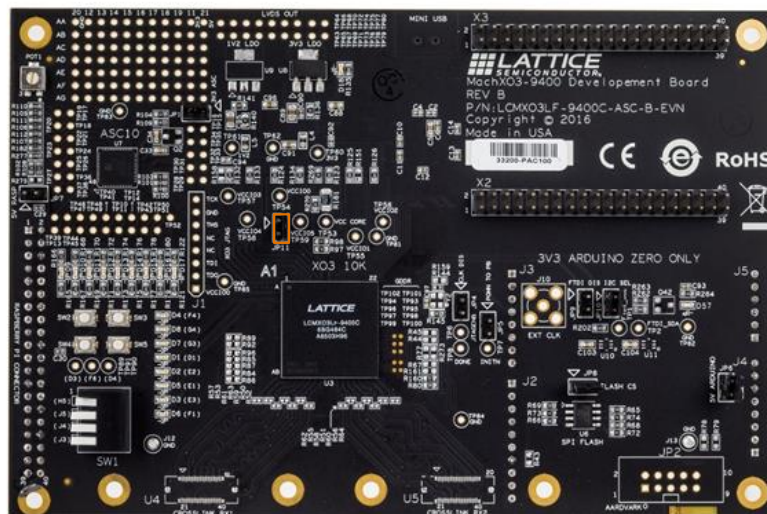


Figure 4.1. MachXO3-9400 Development Board Jumper Settings

Table 4.1. MachXO3-9400 Development Board Jumper Settings

Jumper	Description	Settings
JP11	12 MHz Oscillator	Default short (OSC connected). Alternate open (OSC unconnected).
—	—	All other jumpers should be kept open.

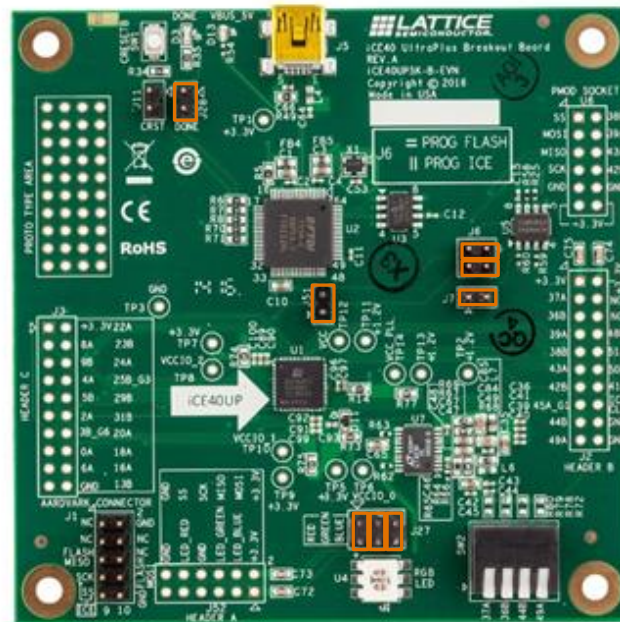


Figure 4.2. iCE40 UltraPlus Breakout Board Jumper Settings

Table 4.2. iCE40 UltraPlus Breakout Board Jumper Settings

Jumper	Description	Settings
J28	DONE LED	Default short
J51	12 MHz Oscillator	Default short (OSC connected). Alternate open (OSC unconnected)
J6	Program SPI Flash or iCE40UP	Both jumpers shorted horizontally
J7	Isolate SPI Flash CSn	Default short
J27	RGB Shunts	All jumpers shorted vertically

5. Programming the Boards

5.1. MachXO3-9400 Development Board (I²C Master)

To program the MachXO3-9400 development board:

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

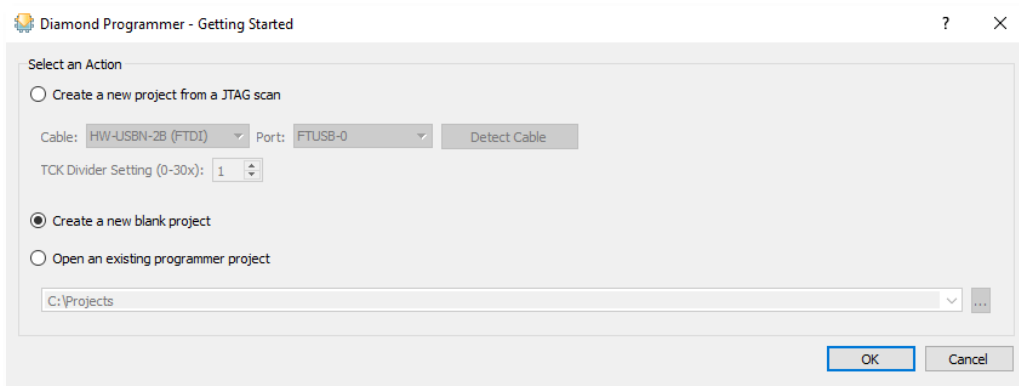


Figure 5.1. Create a New Blank Project

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

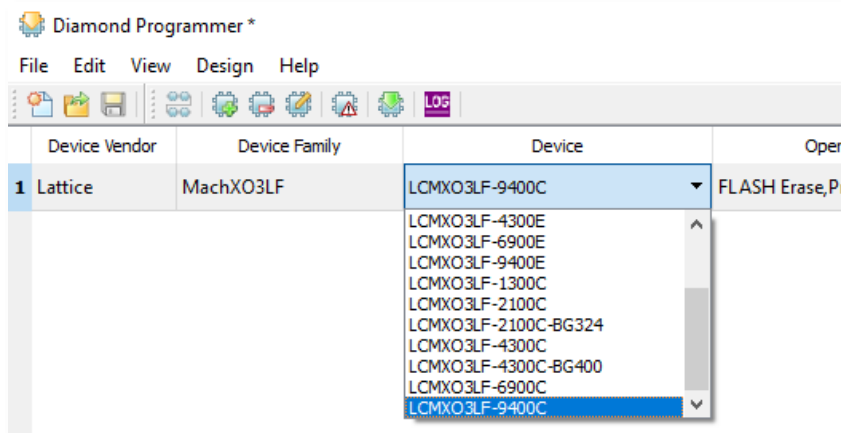


Figure 5.2. Device Selection

3. Right-click and select **Device Properties**.
4. Apply the settings below:
 - **Access mode** – Flash Programming Mode
 - **Operation** – FLASH Erase, Program, Verify
 - **Programming file** – Select the *I2C_Master-XO3.jed* file from the Bitstream folder.

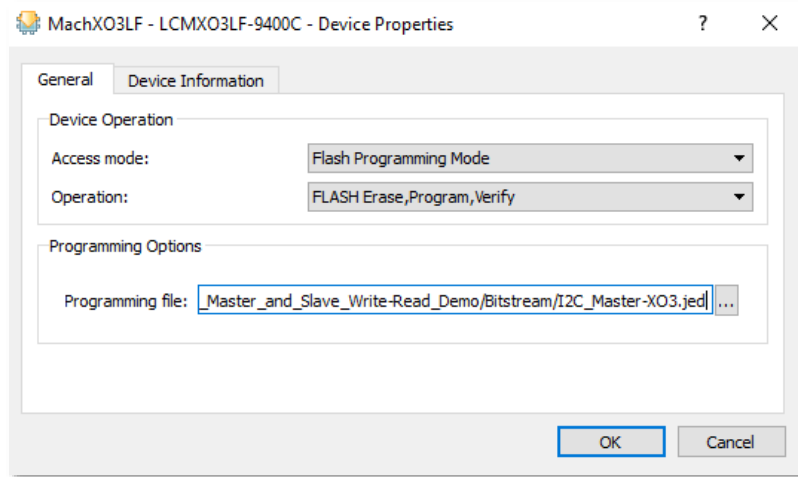



Figure 5.3. Device Properties

5. Click **OK** to close the Device Properties window.
6. Click the **Program** button  in Diamond Programmer to start the Programming sequence.
7. Once programming is successful, the output console displays the message shown in [Figure 5.4](#). If programming fails, press **Detect Cable** in the right pane of the programmer.

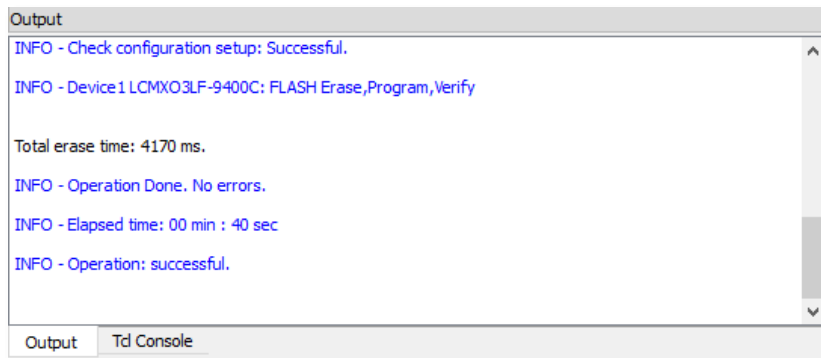


Figure 5.4. Output Console

5.2. iCE40 UltraPlus Breakout Board (I²C Slave)

To program the iCE40 UltraPlus breakout board:

1. Launch the Radiant Programmer with **Create a new blank project**.

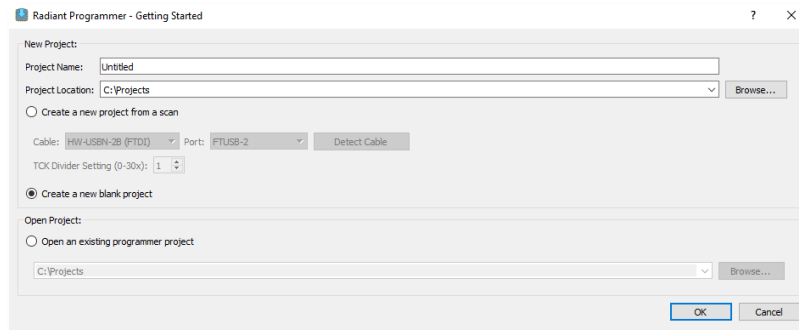


Figure 5.5. Create a New Blank Project

2. Select **iCE40 UltraPlus** for **Device Family** and **iCE40UP5K** for **Device**.

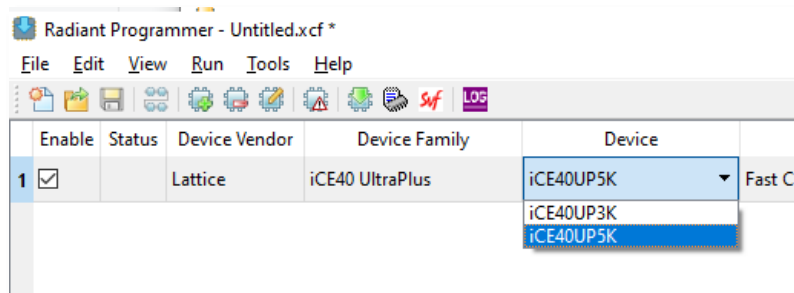


Figure 5.6. Device Selection

3. Right-click and select **Device Properties**.
4. Apply the settings below:
 - **Target Memory** – External SPI Flash Memory (SPI FLASH)
 - **Port Interface** – SPI
 - **Access Mode** – Direct Programming
 - **Operation** – Erase, Program, Verify mode
 - **Programming File** – Select the *I2C_Slave-Ultraplus.jed* file from the Bitstream folder.
 - **SPI Flash Options** – Select the correct Flash chip as shown in Figure 5.7.
 - **Load from File** – Click to refresh fields such as *Data file size* and *End address (Hex)*.

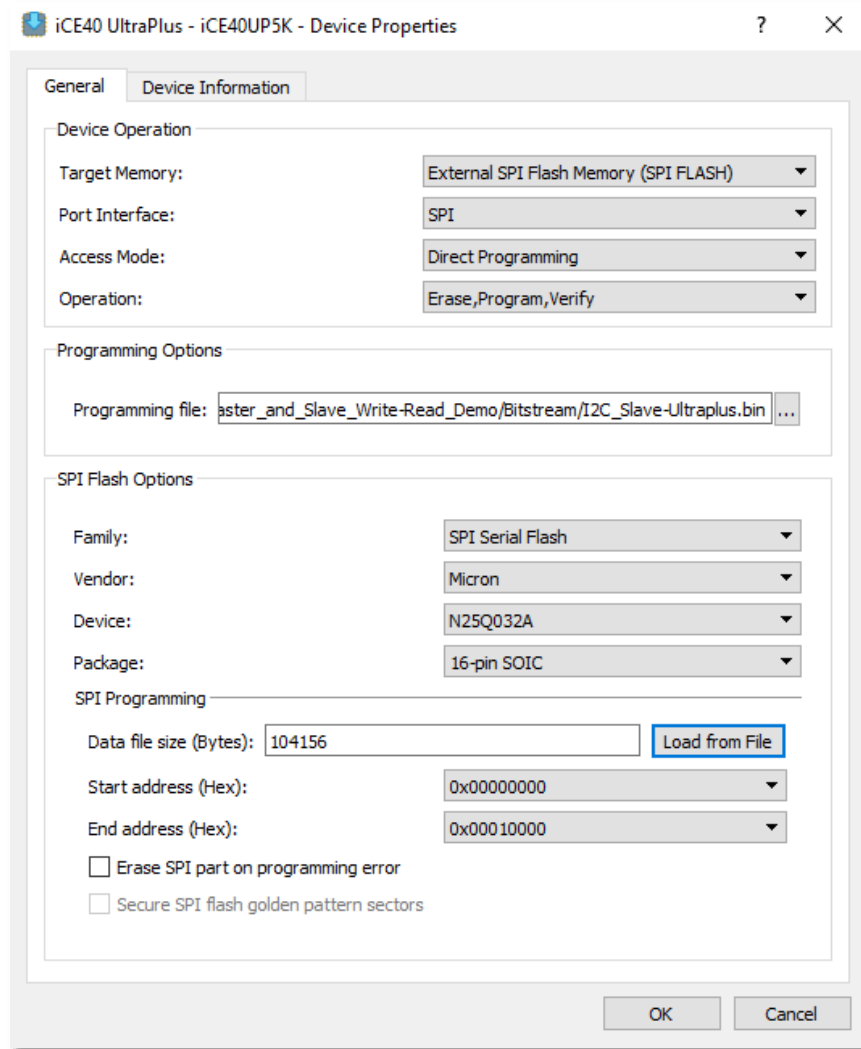



Figure 5.7. Device Properties

5. Click **OK** to close the Device Properties window.
6. Click the **Program** button  in Diamond Programmer to start the Programming sequence.
7. Once programming is successful, the output console displays the message shown in [Figure 5.8](#). If programming fails, select **Detect Cable** in the right pane of the programmer.

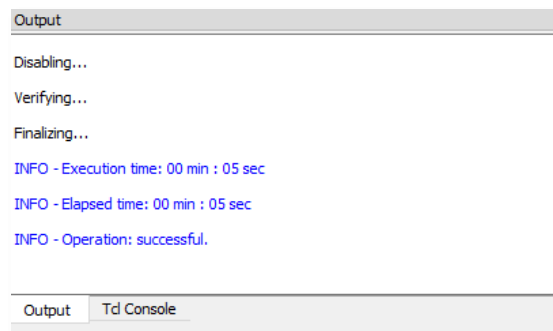


Figure 5.8. Output Console

6. Demo Procedure

To perform the demo:

1. Connect the I²C Ports (SCL and SDA) of the two boards using jumper wires.

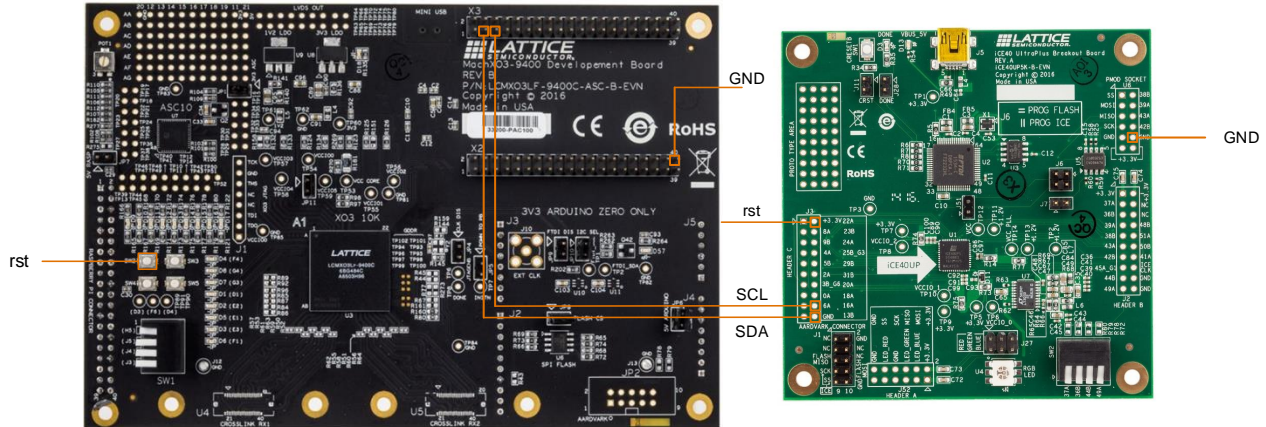


Figure 6.1. SCL and SDA Connections between the Master and the Slave

2. Connect the logic analyser probes to the I²C ports. Ensure that the boards and the logic analyser have common ground connections.
3. Power up the boards by using the USB cables.
4. Assert the Reset (rst) signals of the boards shown in Figure 6.1.
 - a. For the iCE40 UltraPlus board, quickly connect the reset port to a ground pin using a jumper wire and release immediately.
 - b. For the MachXO3 board, quickly press SW5.
5. Using Figure 6.2 and Table 6.1 as a guide, put ADR_MODE and ADR_OPT switches to 0 (down) so that the master can access slave address 0x41 (0b1000001).
6. Using Figure 6.2 and Table 6.2 as a guide, select the desired transaction type and press **BEGIN** until an I²C transaction is registered in the logic analyser as shown in Figure 6.3, Figure 6.4, or Figure 6.5.

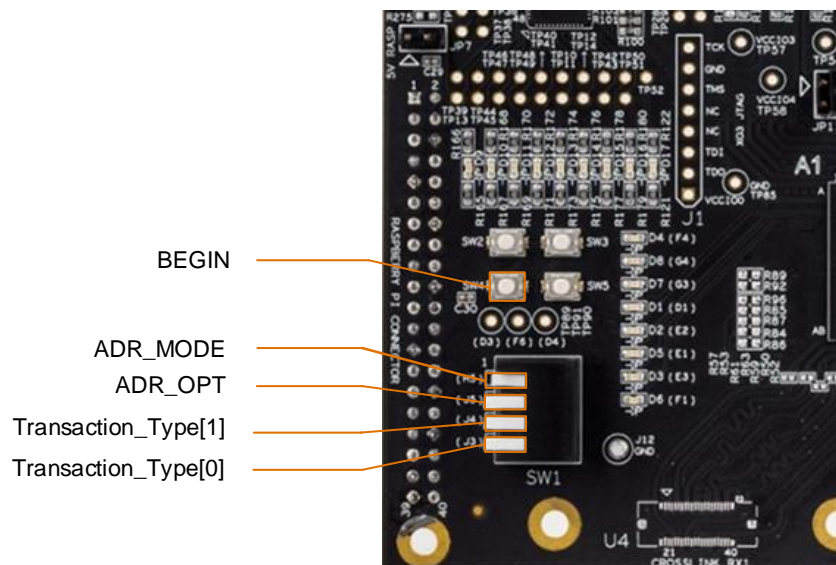


Figure 6.2. Slave Address and Transaction Settings

Table 6.1. Slave Address Options

ADR_MODE	ADR_OPT	Slave Address to be Accessed
0	0	0x41 (0b1000001)
1	0	0x43 (0b1000001)
0	1	0x3C3 (0b1111000001)
1	1	0x3C3 (0b1111000001)

Table 6.2. Slave Address Options

Transaction_Type[1]	Transaction_Type[0]	Slave Address to be Accessed
0	0	Read 3 Bytes of Data
0	1	Write 4 Bytes of Data
1	0	Write 2 Bytes of Data Twice (With Repeated Start)

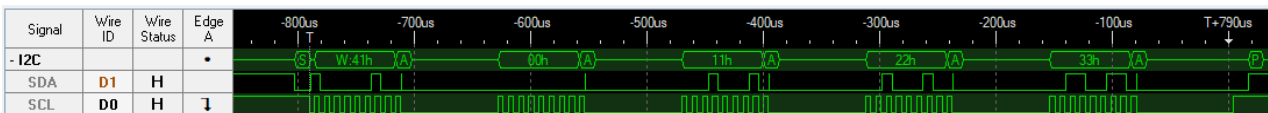


Figure 6.3. I²C Write Transaction

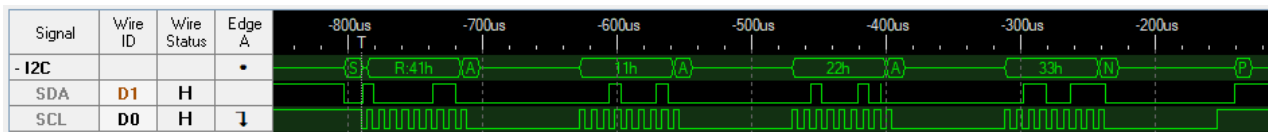


Figure 6.4. I²C Read Transaction

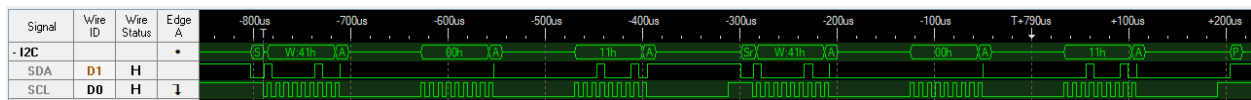


Figure 6.5. I²C Write Transaction with Repeat Start

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.
Device Selection	ECP3	Uncomment only one to enable the selected device.
	ECP5	
	LIFMD	
	LIFCL	
	XO2	
	XO3	
	Ultraplus	
Slave Addresses	SLAVE_ADDRESS1	Define two 10-bit slave addresses that would be accessed by the Master. If 10-bit address mode is disabled, then the controller takes only SLAVE_ADDRESSX[6:0]. Slave demo project always takes SLAVE_ADDRESS1 when compiled in Diamond or Radiant.
	SLAVE_ADDRESS2	
Slave Addresses Mode	ADDR_10BIT_ENABLE	Defines what address mode is taken by the slave project when compiled in Diamond or Radiant. 1'b0 – 7-bit Address Mode 1'b1 – 10-bit Address Mode
Clock Selection	EXT_CLK	Uncomment if external clock is desired. If internal clock is selected, only select 24 MHz in the Clock Speed Selection.
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 have to manually modify the clock dividers of the I ² C Master Controller to allow proper generation of the SCL clock signal.
	CLK_24MHZ	
	CLK_32MHZ	
Clock Stretching Test	stretch_value	Define the duration of the stretch in decimal value.
I ² C Mode Selection	STD	Standard Mode - The testbench I ² C Master generates an SCL clock speed of about 100 KHz.
	FSTMD	Fast Mode - The testbench I ² C Master generates an SCL clock speed of about 400 KHz.

```

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 ECPS
14 //`define LIFMD
15 //`define LIFCL
16 //`define XO2
17 `define XO3
18 //`define Ultraplus
19
20
21 // *****
22 // Slave Addresses
23 // (Define 10-bit slave addresses to be accessed by the master)
24 // *****
25 `define SLAVE_ADDRESS1      10'b111_100_0001 // 0x3C1 or 0x41 depending on whether 10-bit address mode is
26 `define SLAVE_ADDRESS2      10'b111_100_0011 // 0x3C3 or 0x43 depending on whether 10-bit address mode is
27
28 // *****
29 // Slave Addresses Mode
30 // (Define 10-bit slave addresses to be accessed by the master)
31 // *****
32 `define ADDR_10BIT_ENABLE    1'b0 // 1'b1 = 10-Bit Address Mode, 1'b0 = 7-Bit Address Mode
33
34
35
36 // *****
37 // Clock Selection
38 // (Uncomment to enable external clock.)
39 // *****
40 //`define EXT_CLK // Only select 24MHz in the Clock Speed Selection when using the internal oscillator.
41
42
43 // *****
44 // Clock Speed Selection
45 // (Uncomment the selected clock speed.)
46 // *****
47 //`define CLK_12MHE
48 `define CLK_24MHZ
49 //`define CLK_32MHE
50
51
52 // *****
53 // Clock Stretching Test
54 // (Uncomment to enable clock stretching test.)
55 // *****
56 //`define stretch_test

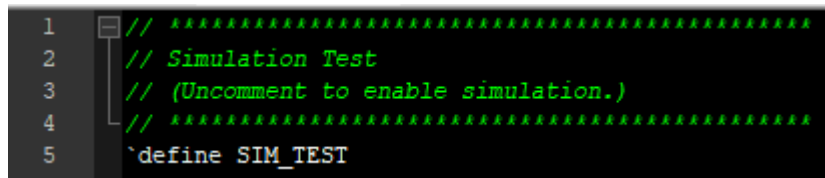
```

Figure 7.1. Compiler Directives Customization Example

Appendix A. Simulation Procedures Using the DO file

The simulation DO file uses both the Soft I²C Master Controller and Soft I²C Slave Peripheral Reference design RTLs in a single Aldec Active-HDL project. 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.



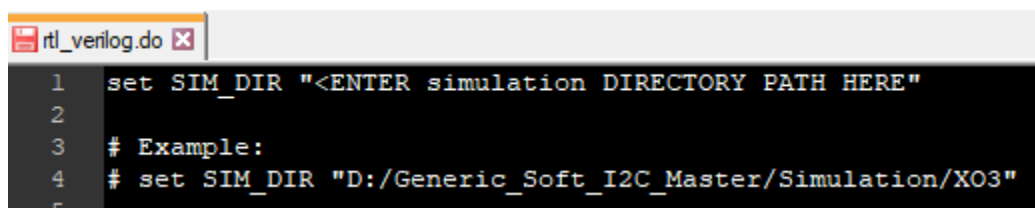
```

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 shown in Line 4 of Figure A.2.



```

1 set SIM_DIR "<ENTER simulation DIRECTORY PATH HERE>"
2
3 # Example:
4 # set SIM_DIR "D:/Generic_Soft_I2C_Master/Simulation/X03"
5

```

Figure A.2. Changing the Simulation Directory

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

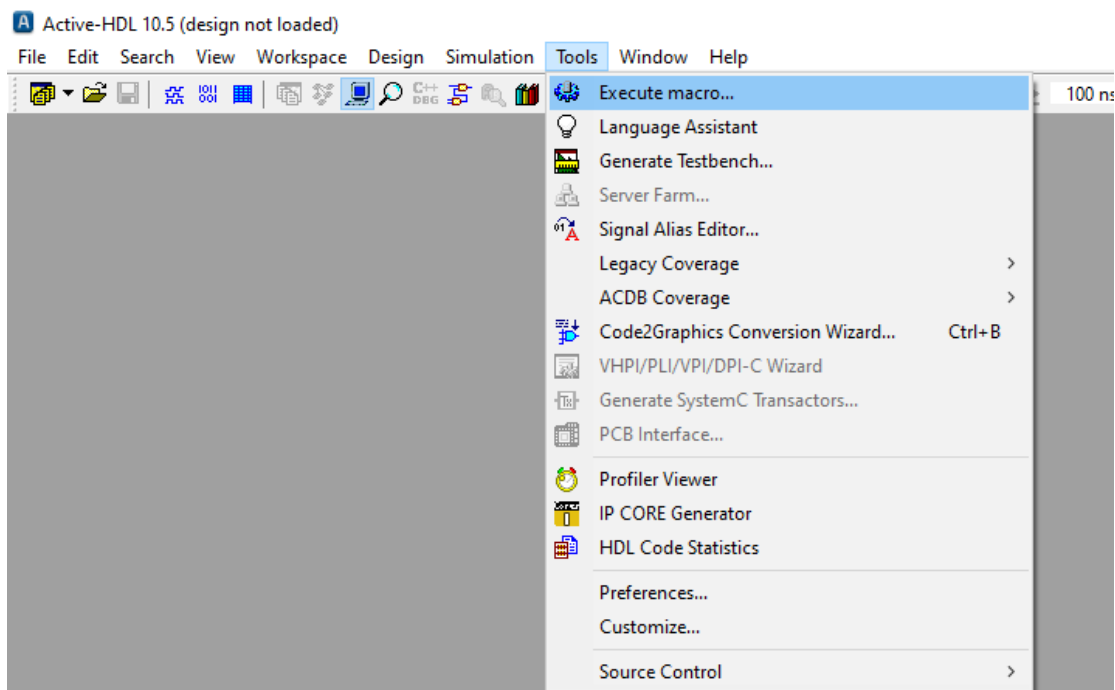


Figure A.3. Running the Simulation File

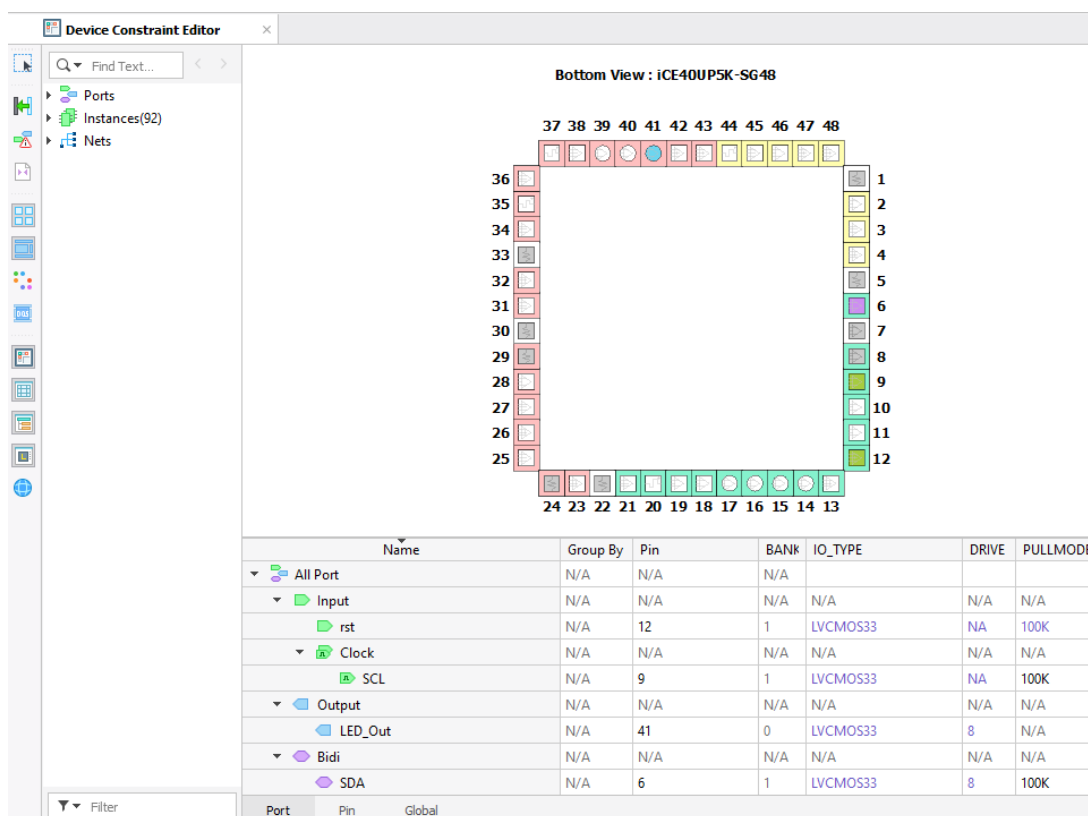
Appendix B. Port Assignments

The projects included in this demo have no port assignments and you should manually assign them based on their selected device and package option. However, the included demo bitstreams (I2C_Master-XO3.jed and I2C_Slave-Ultraplus.bin) are compiled using the port assignments and settings shown in Figure B.1 and Figure B.2.



	Name	Group By	Pin	BANK	BANK_VCC	VREF	IO_TYPE	PULLMODE	DRIVE	SLEWRATE	CLAMP
1	All Ports	N/A	N/A	N/A	N/A	N/A	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	N/A	N/A	N/A	N/A
1.1.1	ADR_MODE	N/A	H5(H5)	5(5)	Auto	N/A	LVC MOS3...	UP(UP)	NA(NA)	NA(NA)	ON(ON)
1.1.2	ADR_OPT	N/A	J5(J5)	5(5)	Auto	N/A	LVC MOS3...	UP(UP)	NA(NA)	NA(NA)	ON(ON)
1.1.3	BEGIN	N/A	F6(F6)	5(5)	Auto	N/A	LVC MOS3...	UP(UP)	NA(NA)	NA(NA)	ON(ON)
1.1.4	Transaction_Type[0]	N/A	J3(J3)	5(5)	Auto	N/A	LVC MOS3...	UP(UP)	NA(NA)	NA(NA)	ON(ON)
1.1.5	Transaction_Type[1]	N/A	J4(J4)	5(5)	Auto	N/A	LVC MOS3...	UP(UP)	NA(NA)	NA(NA)	ON(ON)
1.1.6	rst	N/A	D3(D3)	5(5)	Auto	N/A	LVC MOS3...	UP(UP)	NA(NA)	NA(NA)	ON(ON)
1.2	Output	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.2.1	LED_Out	N/A	F4(F4)	5(5)	Auto	N/A	LVC MOS3...	NONE(NONE)	8(8)	SLOW(SLOW)	OFF(OFF)
1.3	Bidir	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.3.1	SCL	N/A	F9(F9)	0(0)	Auto	N/A	LVC MOS3...	UP(UP)	8(8)	SLOW(SLOW)	ON(ON)
1.3.2	SDA	N/A	F8(F8)	0(0)	Auto	N/A	LVC MOS3...	UP(UP)	8(8)	SLOW(SLOW)	ON(ON)
1.4	Others	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.4.1	Nonexistent	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
1.4.1.1	clk	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

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



Bottom View : iCE40UP5K-SG48

37 38 39 40 41 42 43 44 45 46 47 48

36 35 34 33 32 31 30 29 28 27 26 25

24 23 22 21 20 19 18 17 16 15 14 13

1 2 3 4 5 6 7 8 9 10 11 12

Name	Group By	Pin	BANK	IO_TYPE	DRIVE	PULLMODE
All Port	N/A	N/A	N/A	N/A	N/A	N/A
Input	N/A	N/A	N/A	N/A	N/A	N/A
rst	N/A	12	1	LVC MOS33	NA	100K
Clock	N/A	N/A	N/A	N/A	N/A	N/A
SCL	N/A	9	1	LVC MOS33	NA	100K
Output	N/A	N/A	N/A	N/A	N/A	N/A
LED_Out	N/A	41	0	LVC MOS33	8	N/A
Bidi	N/A	N/A	N/A	N/A	N/A	N/A
SDA	N/A	6	1	LVC MOS33	8	100K

Port Pin Global

Figure B.2. iCE40 UltraPlus 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 \(FPGA-DS-02007\)](#)
- [MachXO2 Family Data Sheet \(DS1035\)](#)
- [MachXO3 Family Data Sheet \(FPGA-DS-02032\)](#)
- [iCE40 UltraPlus Family Data Sheet \(FPGA-DS-02008\)](#)
- [Generic Soft I²C Master Controller \(FPGA-RD-02201\)](#)
- [Generic Soft I²C Master Slave Peripheral Reference Design \(FPGA-RD-02193\)](#)

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