Lattice Avant Device Multi-Boot

Reference Design

FPGA-RD-02321-1.0

April 2026

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

© 2026 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.

Contents

Contents 3
Abbreviations in This Document 5
1. Introduction 6
1.1. Quick Facts 6
1.2. Feature 6
1.3. Naming Conventions 6
1.3.1. Nomenclature 6
1.3.2. Signal Names 6
2. Directory Structure and Files 7
3. Functional Description 8
3.1. Design Components 8
4. Reference Design Parameter Description 11
5. Signal Description 12
6. Running the Reference Design 13
6.1. Compiling the Reference Design 13
6.2. Generating the Bitstream File 13
6.3. Generating Initial Multi-Boot Programming File 15
7. Implementing the Reference Design on Board 20
7.1. Hardware Requirements 20
7.2. Programming .mcs into External SPI Flash 20
7.3. Enabling 32-bit MSPI Address and Command Mode 20
7.3.1. Setting 32-bit MSPI Address and Command in Non-Volatile Configuration Memory Using Pseudo Programming Mode 21
7.3.2. Setting 32-bit MSPI Address and Command in Non-Volatile Configuration Memory Using Direct Programming Mode 24
7.4. Running the Design 24
7.4.1. Performing Multi-Boot – without Corrupted Pattern 24
7.4.2. Performing Multi-Boot – with Corrupted Pattern 26
References 28
Technical Support Assistance 29
Revision History 30

Figures

Figure 2.1. Directory Structure 7
Figure 3.1. Reference Design Block Diagram 8
Figure 3.2. State Diagram of Refresh Controller Instance 10
Figure 6.1. Lattice Radiant Software 13
Figure 6.2. Open Project File 14
Figure 6.3. Generated Bitstream Log 14
Figure 6.4. Launch Radiant Deployment Tool from Radiant Programmer 15
Figure 6.5. Radiant Deployment Tool – Getting Started 15
Figure 6.6. Step 1 of 4: Select Input File(s) 16
Figure 6.7. Options Tab of Step 2 of 4: Advanced SPI Flash Option 17
Figure 6.8. Multiple Boot Tab of Step 2 of 4: Advanced SPI Flash Option 18
Figure 6.9. Step 3 of 4: Select Output File(s) 18
Figure 6.10. Step 4 of 4: Generate Deployment 19
Figure 7.1. Device Properties Window 20
Figure 7.2. Device Properties Window for Feature Row Pseudo Programming 21
Figure 7.3. Feature Row Window 22

© 2026 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.

Figure 7.4. Change the Values of 32-bit SPIM Address and 32-bit SPIM Command 22
Figure 7.5. Confirm Overwrite Register 23
Figure 7.6. Output Window Showing Operation: Successful 23
Figure 7.7. multiboot.rva File 24
Figure 7.8. apps_image Showing CAFE0 25
Figure 7.9. apps_image Showing CAFE1 25
Figure 7.10. apps_image Showing CAFE2 25
Figure 7.11. Reprogram Corrupted Image 26
Figure 7.12. apps_image Showing CAFE0 27

Tables

Table 1.1. Summary of the Reference Design 6
Table 2.1. File List 7
Table 4.1. Parameters in cmd_list.v 11
Table 5.1. Primary I/O 12

© 2026 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.

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
FPGA	Field Programmable Gate Array
HDL	Hardware Description Language
IP	Intellectual Property
LED	Light-Emitting Diode
LMMI	Lattice Memory Mapped Interface
OTP	One-Time Programmable
RTL	Register Transfer Level
SPI	Serial Peripheral Interface
USB	Universal Serial Bus

© 2026 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.

1. Introduction

This reference design showcases the Multi-Boot mode supported in Lattice Avant™ devices. The Multi-Boot mode supports booting from up to 18 patterns that reside in an external SPI flash device. The patterns include a primary pattern, a golden pattern, and up to 16 alternate patterns, designated as Alternate Pattern 1 to Alternate Pattern 16.

The device boots by loading the primary pattern from the external flash. In static mode, toggling the PROGRAMN pin or receiving a REFRESH command always loads Alternate Pattern 1. Each subsequent PROGRAMN/REFRESH event loads the next pattern defined in the Multi-Boot configuration. If the loading of the primary pattern or any alternate pattern fails, the device attempts to load the golden pattern. The bitstream pattern sequence, target address of the golden pattern, and target addresses of the alternate patterns are defined during the Multi-Boot configuration process in the Lattice Radiant™ Deployment Tool.

Using the CONFIG_LMMIC primitive allows the device to operate in the dynamic mode. The dynamic mode allows the system to dynamically switch to any of the alternate patterns after the device boots up from the primary pattern while still being protected by a golden pattern. This reference design implements the dynamic mode, which allows the system to dynamically switch between two and three bitstream patterns using the CONFIG_LMMIC primitive. By using the
Multi-Boot mode, you can combine all the bitstream patterns into a single bitstream image and store it in a single external SPI flash device. This solution decreases cost, reduces board space, and simplifies field upgrades. Note that this reference design is developed using the Avant-X70 device, but the design can be ported to other Avant devices.

Refer to Lattice Avant Multi-Boot User Guide (FPGA-TN-02314) for more information.

1.1. Quick Facts

Download the reference design files from the Lattice reference design web page.

Table 1.1. Summary of the Reference Design

General	Target Devices	Lattice Avant-X70 (LAV-AT-X70-3LFG1156I)
	Source Code Format	Verilog
Simulation	Functional Simulation	Not performed
	Timing Simulation	Not performed
	Test Bench	Not available
	Test Bench Format	Not available
Software Requirements	Software Tool and Version	2024.2 SP1
	IP Version (if applicable)	OSC 2.1.0
Hardware Requirements	Board	Avant-G/X Versa Board
	Cable	USB-A to Mini-B Programming Cable

1.2. Feature

The key feature of the Multi-Boot reference design is that it allows the system to dynamically switch between two and up to three bitstream patterns using the CONFIG_LMMIC primitive while still being protected with a golden pattern.

1.3. Naming Conventions

1.3.1. Nomenclature

The nomenclature used in this document is based on Verilog HDL.

1.3.2. Signal Names

- *_n* are active-low signals, asserted when the value is logic 0.
- *_i* are input signals.
- *_o* are output signals.

© 2026 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.

2. Directory Structure and Files

Figure 2.1 shows the directory structure.

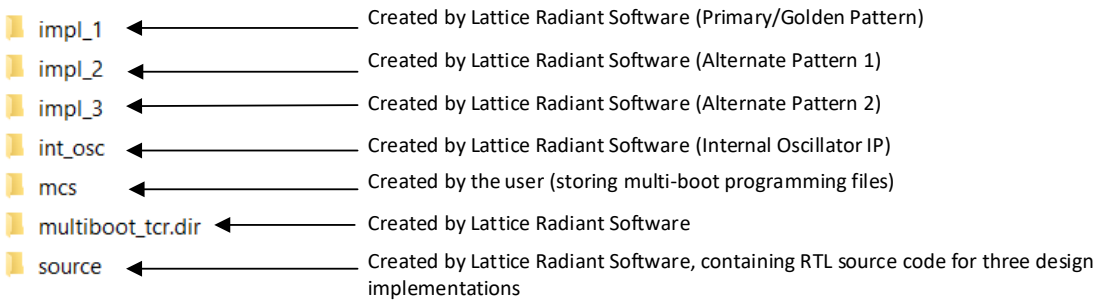

Figure 2.1. Directory Structure

Table 2.1 shows the list of files included in the reference design package.

Table 2.1. File List

Attribute	Description
<Component name>.ipx	This file contains the information on the files associated with the generated IP.
<Component name>.cfg	This file contains the parameter values used in the IP configuration.
component.xml	Contains the ipxact: component information of the IP.
design.xml	Documents the configuration parameters of the IP in IP-XACT 2014 format.
rtl/<Component name>.v	This file provides an example RTL top file that instantiates the module.
rtl/<Component name>_bb.v	This file provides the synthesis closed box.
misc/<Component name>_tmpl.v misc /<Component name>_tmpl.vhd	These files provide instance templates for the module.

© 2026 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.

3. Functional Description

Figure 3.1 shows the top-level block diagram of the reference design. The blocks shown in Figure 3.1 are the fundamental blocks that appear in all the four design implementations for Multi-Boot operations.

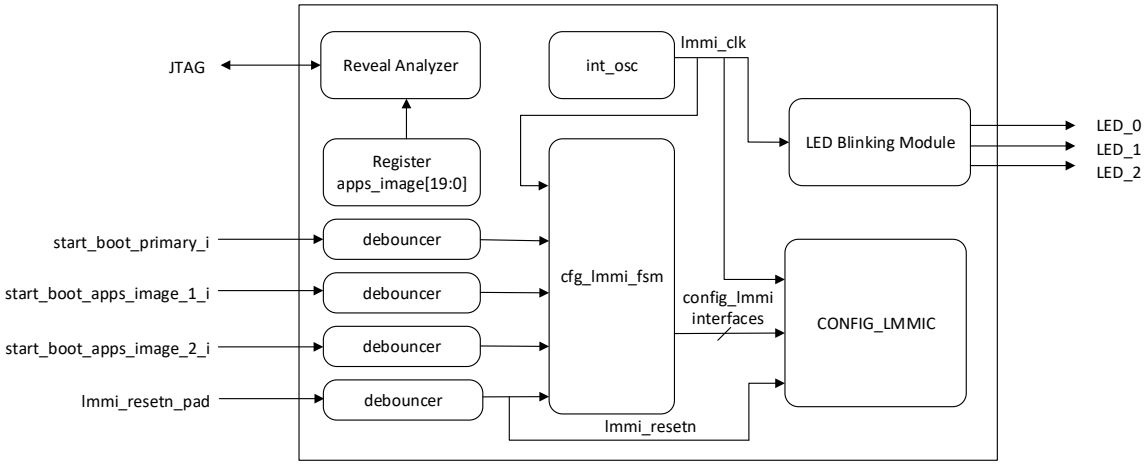

Figure 3.1. Reference Design Block Diagram

3.1. Design Components

The Lattice Avant Multi-Boot reference design includes the following blocks:

- Int_osc: This is an oscillator module. It generates the clock source (lmmi_clk) to the cfg_lmmi_fsm module, CONFIG_LMMIC primitive, and the LED blinking module. The lmmi_clk clock is running at 25 MHz. Refer to Lattice Avant OSC Module User Guide (FPGA-IPUG-02184) for more information.
- LED blinking module: Implements a counter to blink the three LEDs on board.
- Reveal Analyzer: Reveal Analyzer is used to determine which alternate pattern is running on the device.
- Debouncer: This block eliminates the noise or fluctuations, also known as bouncing, in a digital input signal, which can occur when a mechanical switch is pressed or released.
- CONFIG_LMMIC: This is the Lattice Memory Mapped Interface (LMMI) interface to the configuration block. Refer to the CONFIG_LMMIC section in Lattice Radiant Software Help for more information.
- cfg_lmmi_fsm: Implements a state machine controller to send the necessary commands to the CONFIG_LMMIC primitive to boot the desired alternate pattern stored in an external SPI flash. The controller performs the following sequences. Refer to Figure 3.2 for more details.
  - IDLE: Idle and waiting for either one of these input signals (start_boot_primary_i, start_boot_apps_image_1_i, or start_boot_apps_image_2_i) to perform multi-boot operations.
  - LMMI_WRITE_NOOP: Executes 32 bytes of NOOP command.
    - LMMI Offset = 8’h00
    - LMMI_CFG_DATA[15:0] = 16’h0000
  - PORT_REQUEST_FORCE: Executes the port request with force request for exclusive access to the configuration engine.
    - LMMI Offset = 8’h02 and LMMI_CFG_PORT_REQUEST[31:16]=16’h7A00
    - LMMI Offset = 8’h01 and LMMI_CFG_PORT_REQUEST[15:0]=16’h0002
      Note: Write LMMI_CFG_PORT_REQUEST[15:0] first followed by LMMI_CFG_PORT_REQUEST[31:16]. Failure to follow this sequence might cause the LMMI port to become inactive, thus requiring power cycling of the device to recover the LMMI port.
  - PORT_STATUS: Reads out the port status, LMMI_CFG_PORT_STATUS[15:0], for the CONFIG_LMMI interface. Proceed to the next state once the exclusive access is claimed. If the port status is available, go back to the PORT_REQUEST_FORCE state to claim exclusive access. If the port is inactive or disabled, go back to the idle state.

© 2026 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.

    - LMMI Offset = 8’h03
- MSPI_MULTIBOOT_ADDR: Sends the address through the CONFIG_LMMIC primitive. The address refers to the start address where you store the bitstream in the external SPI flash and that is the bitstream or pattern you desire the device to boot for configuring the device.
    - LMMI Offset = 8’h05 and LMMI_CFG_MSPI_MULTIBOOT_ ADDR [15:0] = boot_address_15_to_0
    - LMMI Offset = 8’h06 and LMMI_CFG_MSPI_MULTIBOOT_ ADDR [31:16] = boot_address_31_to_16
- REFRESH Command: Equivalent to toggling the PROGRAMN pin. Once this command is executed, the device starts to load the desired alternate pattern from the external SPI flash according to the boot address sent in the MSPI_MULTIBOOT_ADDR state. If the loading of the image fails, the device falls back to load the golden pattern. In case the REFRESH command is not executed properly, proceed to the next state to release the exclusive access to the configuration engine.
    - LMMI Offset = 8’h00 and LMMI_CFG_DATA[15:0] = 16’h8000
    - LMMI Offset = 8’h00 and LMMI_CFG_DATA[15:0] = 16’h0A00
- PORT_REQUEST_RELEASE: Ensures the flow terminates gracefully by releasing exclusive access to the configuration engine, especially in cases where the previous REFRESH command may not have executed correctly.
    - LMMI Offset = 8’h02 and LMMI_CFG_PORT_REQUEST[31:16]=16’h7A00
    - LMMI Offset = 8’h01 and LMMI_CFG_PORT_REQUEST[15:0]=16’h0003

Refer to Table D.2. LMMI CFG Offset List and Table 6.12. Target Configuration Commands in the Avant sysCONFIG User Guide (FPGA-TN-02299) for more information.

© 2026 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.

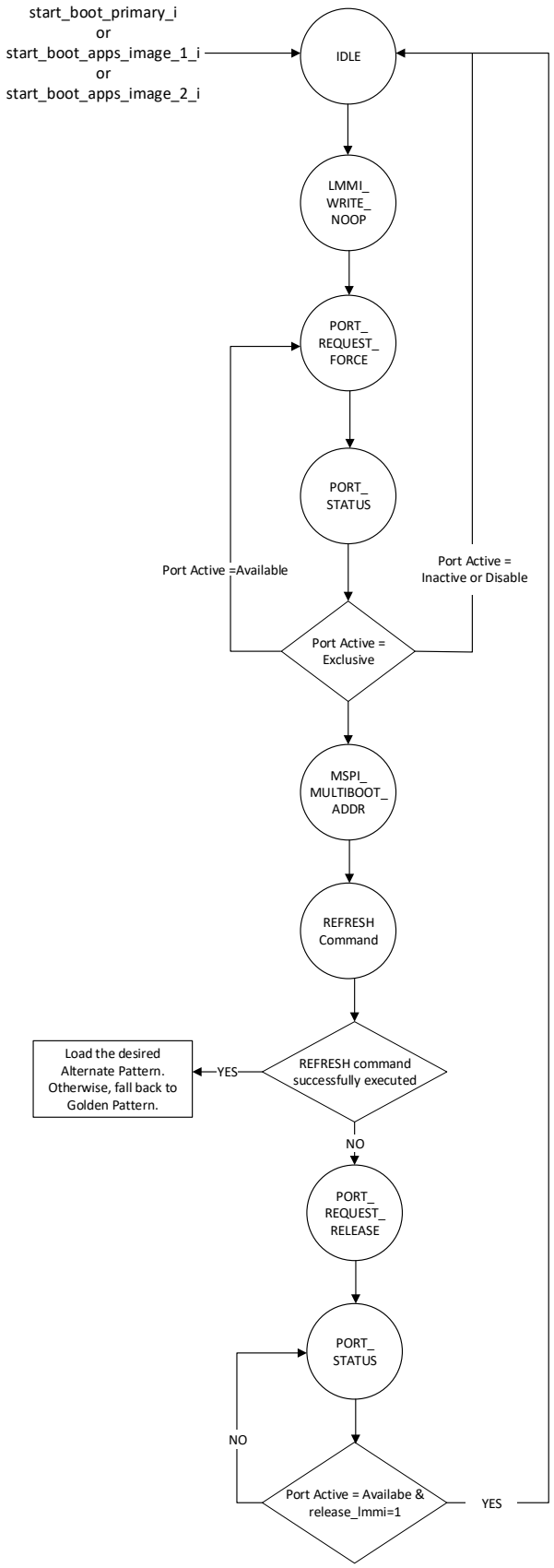

Figure 3.2. State Diagram of Refresh Controller Instance

© 2026 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.

4. Reference Design Parameter Description

The Multi-Boot reference design includes parameters shown in Table 4.1. You can modify the parameters defined in the cmd_list.v file according to the start address where you stored the primary or alternate patterns.

Table 4.1. Parameters in cmd_list.v

Parameter	Default Value	Description
BOOT_ADDRESS_15_TO_0_PRIMARY	16’h0000	Boot address for Primary Pattern. Lower 16 bits of the 32-bit address.
BOOT_ADDRESS_31_TO_16_PRIMARY	16’h0000	Boot address for Primary Pattern. Upper 16 bits of the 32-bit address.
BOOT_ADDRESS_15_TO_0_APPS_IMAGE_1	16’h0000	Boot address for Alternate Pattern 1. Lower 16 bits of the 32-bit address.
BOOT_ADDRESS_31_TO_16_APPS_IMAGE_1	16’h0180	Boot address for Alternate Pattern 1. Upper 16 bits of the 32-bit address.
BOOT_ADDRESS_15_TO_0_APPS_IMAGE_2	16’h0000	Boot address for Alternate Pattern 2. Lower 16 bits of the 32-bit address.
BOOT_ADDRESS_31_TO_16_APPS_IMAGE_2	16’h0240	Boot address for Alternate Pattern 2. Upper 16 bits of the 32-bit address.

© 2026 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.

5. Signal Description

The input/output interface signals for the top.v are shown in Table 5.1.

Table 5.1. Primary I/O

Port Name	I/O	Width	Description
LED_0	Out	1	Output to blink LED D45
LED_1	Out	1	Output to blink LED D46
LED_2	Out	1	Output to blink LED D47
start_boot_primary_i	In	1	Active low input port. Toggle the SW13 push button to trigger the boot for primary pattern.
start_boot_apps_image_1_i	In	1	Active low input port. Toggle the SW14 push button to trigger the boot for Alternate Pattern 1.
start_boot_apps_image_2_i	In	1	Active low input port. Toggle the SW15 push button to trigger the boot for Alternate Pattern 2.
lmmi_resetn_pad	In	1	An active low asynchronous reset port. Toggle the SW12 push button to assert the reset.

© 2026 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.

6. Running the Reference Design

This section describes how to run the Multi-Boot reference design using the Lattice Radiant software. For more details on the Lattice Radiant software, refer to the Lattice Radiant Software User Guide.

The project consists of three design implementations that can be used to distinguish the pattern running on the device. They differ in a unique 20-bit apps_image register value and the output port(s) to blink LED(s) on board. Below is the apps_image value for each design implementation.

- impl_1: apps_image=0xcafe0 and output port to blink one LED (D45)
- impl_2: apps_image=0xcafe1 and output ports to blink two LEDs (D45 and D46)
- impl_3: apps_image=0xcafe2 and output ports to blink three LEDs (D45, D46, and D47)

6.1. Compiling the Reference Design

The reference design file includes the pre-compiled files and bitstreams in .bit and .mcs file formats, to help you get started. However, you can recompile the reference design in newer Lattice Radiant versions after adding or modifying the user logic if you choose to do so. To do that, you can set the design implementation as active implementation, make the desired changes, and recompile the design through the standard compilation flow.

Note: .mcs files are the data record files that are in the format commonly known as Intel Hex, Motorola Hex, or Extended Tektronix Hex. They are also known as addressed record files. The advantages include its small size and its printable feature, which make it good for record-keeping. This type of file is not directly consumable by the utilities supporting it.

6.2. Generating the Bitstream File

This section provides the procedure of creating your FPGA bitstream file using the Lattice Radiant software.
To create the FPGA bitstream file, follow steps below.

1. Open the Lattice Radiant software, as shown in Figure 6.1.

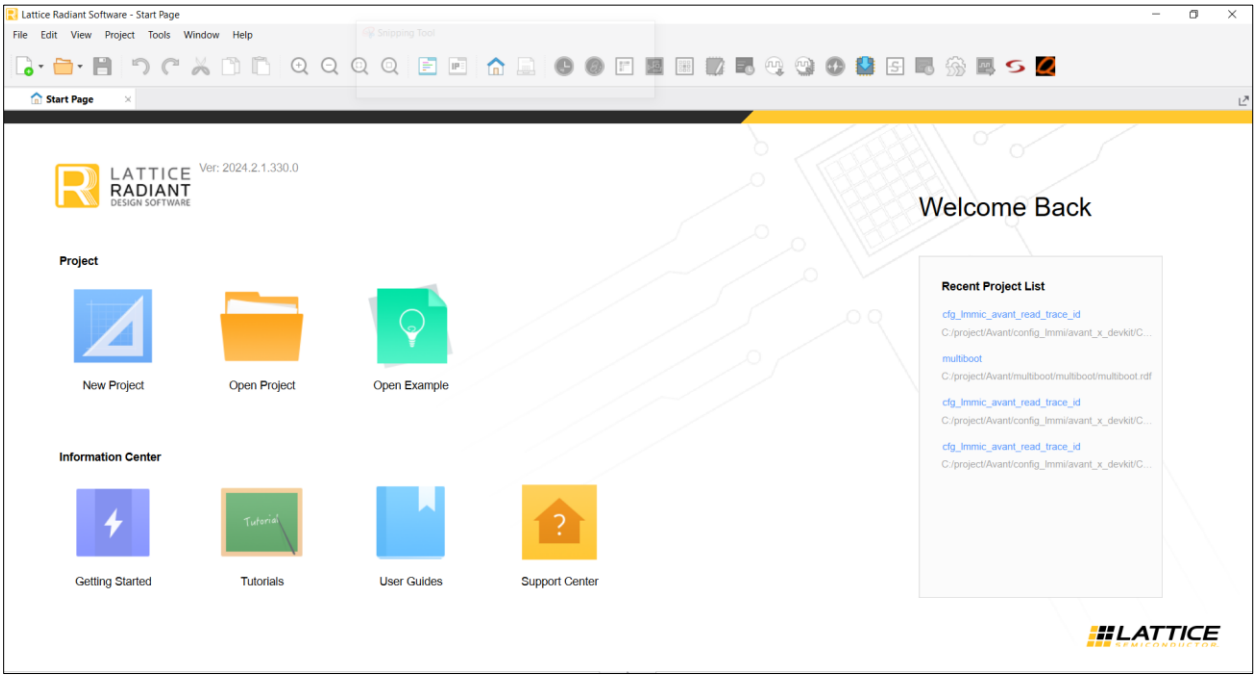

Figure 6.1. Lattice Radiant Software

2. Click **File > Open Project**. From the project database, browse to the reference design project, the multiboot folder and open the Lattice Radiant project file, lmmi2mspi.rdf, as shown in Figure 6.2.

© 2026 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.

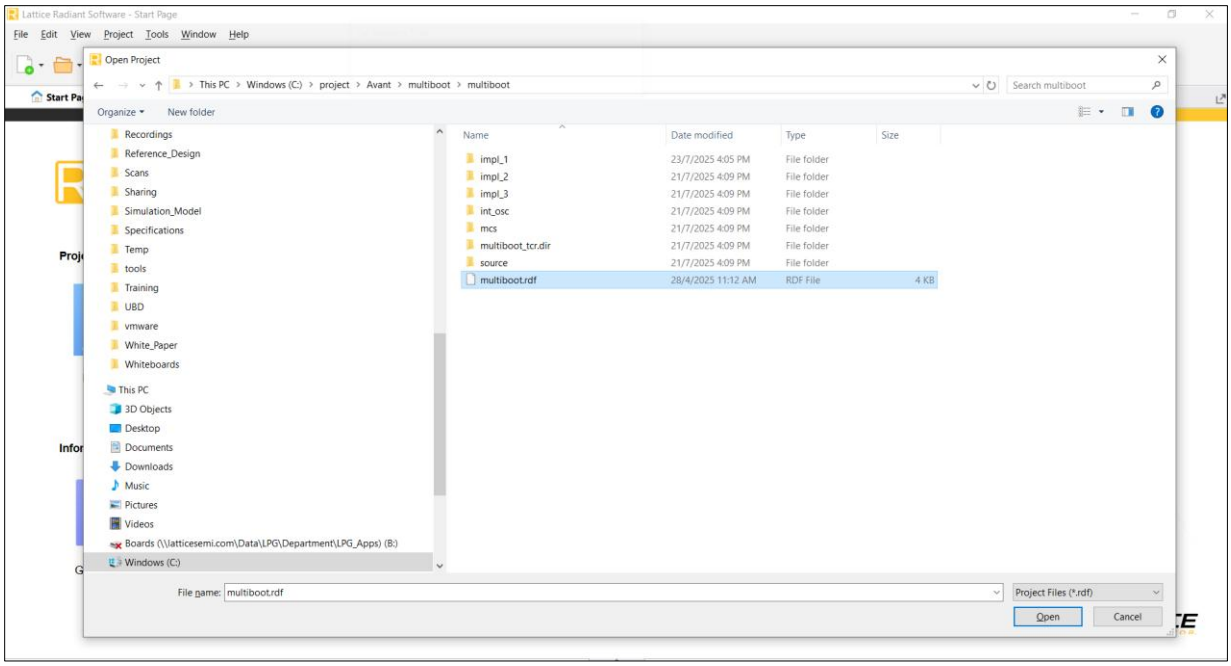

Figure 6.2. Open Project File

3. Click **Export Files** to generate the bit file (Figure 6.3). View the log message in the Export Reports folder for the generated bitstream. Once the compilation is successful, the generated bit file is in the project implementation folder, for example, multiboot\impl_1\multiboot_impl_1.bit. Repeat this step for others design implementations if you need to recompile the design.

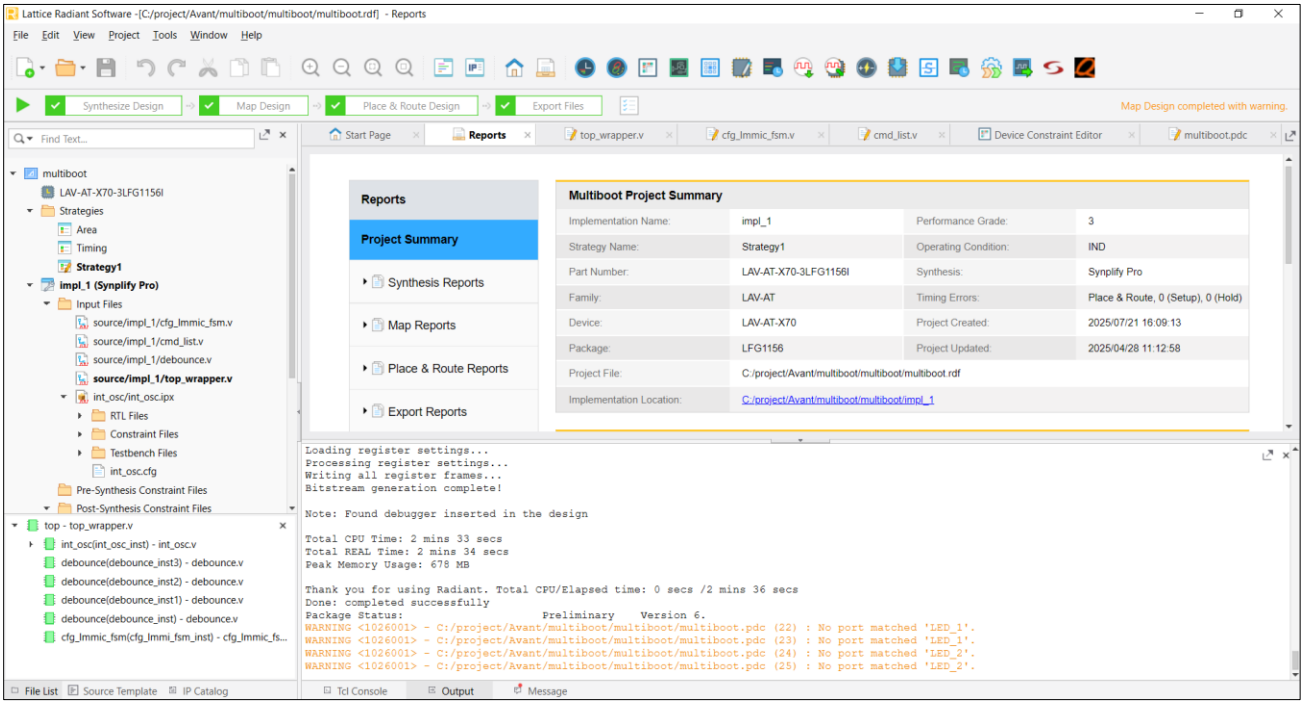

Figure 6.3. Generated Bitstream Log

© 2026 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.

6.3. Generating Initial Multi-Boot Programming File

This section describes how to generate the initial Multi-Boot programming file.
Note: You can use the provided Multi-Boot programming file located in the \multiboot\mcs\multiboot.mcs directory and skip the steps below.

Follow steps below to generate the single Multi-Boot programming file (.mcs).

1. Launch the Radiant Deployment Tool from Radiant Programmer by clicking **Tools** > **Deployment Tool** (Figure 6.4).

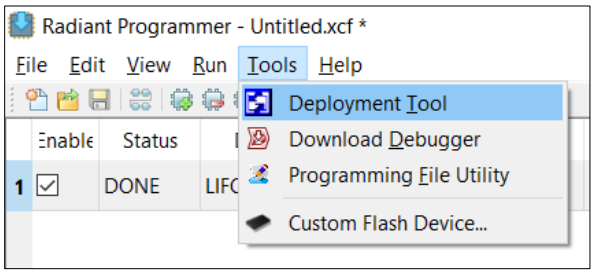

Figure 6.4. Launch Radiant Deployment Tool from Radiant Programmer

2. In the Radiant Deployment Tool window, select External Memory for **Function Type** and Advanced SPI Flash for **Output File Type**. Click **OK** (Figure 6.5).

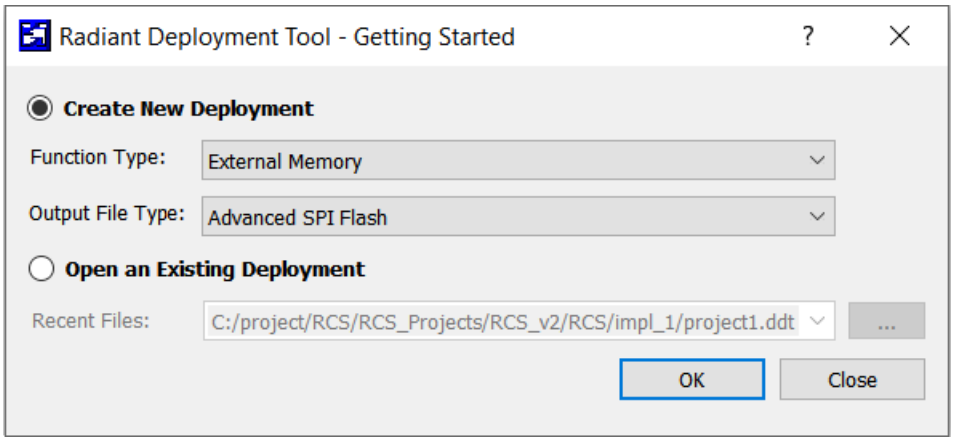

Figure 6.5. Radiant Deployment Tool – Getting Started

3. A four-step wizard opens, showing: External Memory: Advanced SPI Flash.
   a. In Step 1 of 4: Select Input File(s), select a .bit file for **File Name** and click **Next** (Figure 6.6). This is the Primary Image.

© 2026 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.

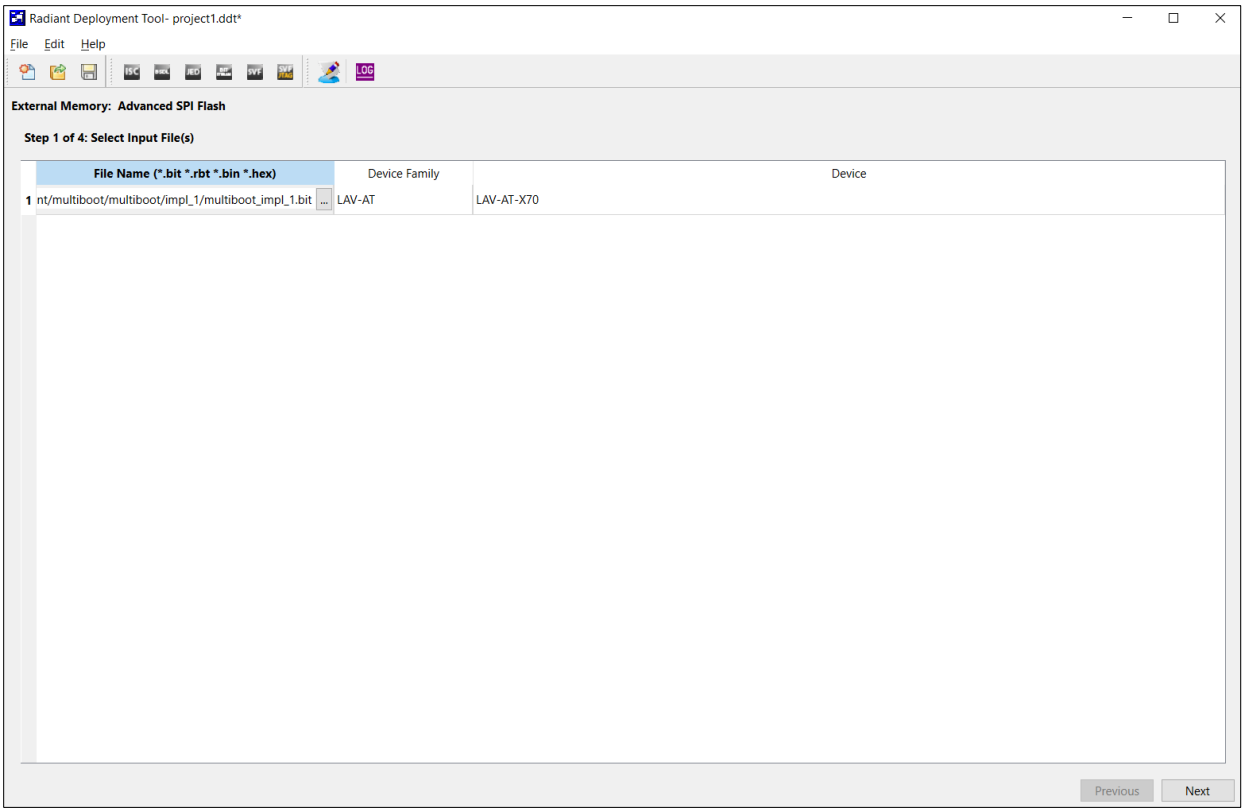

Figure 6.6. Step 1 of 4: Select Input File(s)

b. In Step 2 of 4: Advanced SPI Flash Option, under the **Options** tab, select the following options (Figure 6.7):
   - Output Format: Intel Hex
   - SPI Flash Size (Mb): 512
   - SPI Flash Read Mode: Standard Read
   - Clock Mode: Single-ended Clock

© 2026 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.

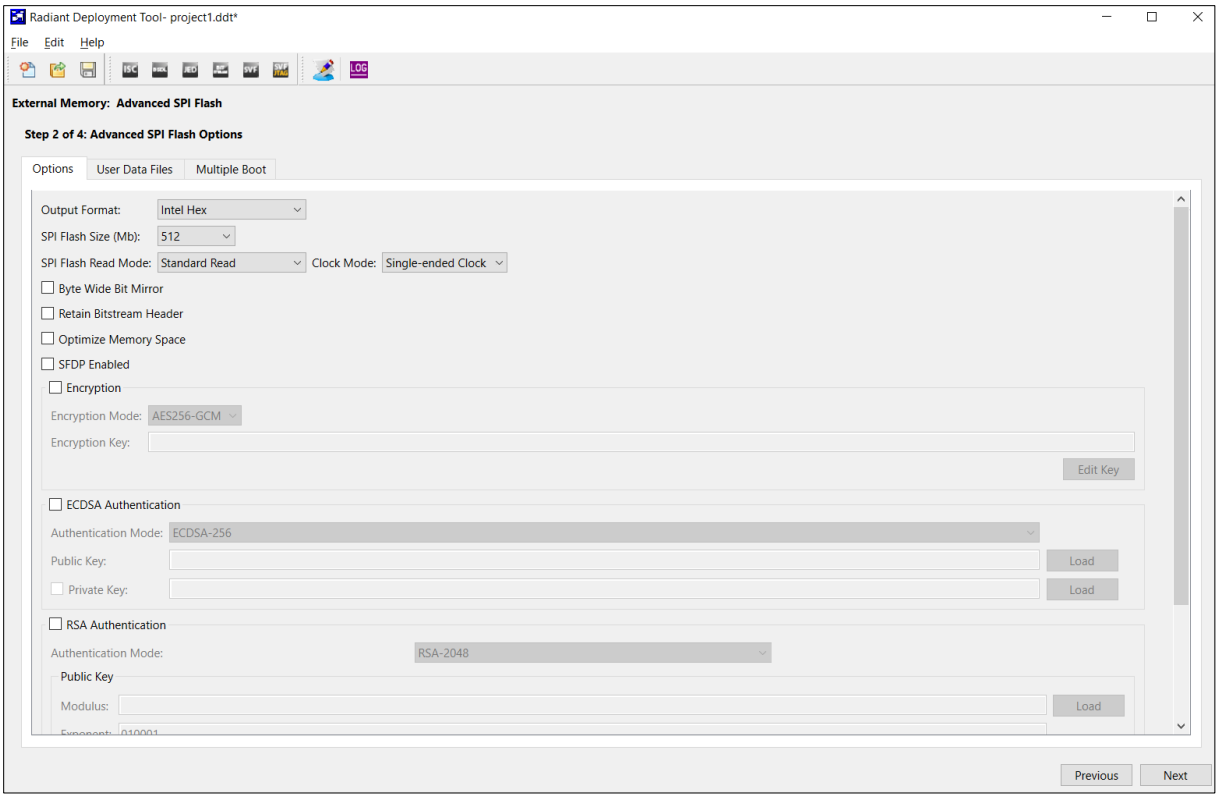

Figure 6.7. Options Tab of Step 2 of 4: Advanced SPI Flash Option

c. In Step 2 of 4: Advanced SPI Flash Option, under the **Multiple Boot** tab, select the following options (Figure 6.8):
   - Check Multiple Boot.
   - Golden Pattern:
     Select the multiboot_impl_1.bit file in the multiboot/impl_1 folder.
   - Starting Address: 0x00C00000.
   - Multi_Boot_Sel Options: Dynamic
   - Number of Alternate Patterns: 2 – to program two alternative images in the external SPI flash.
   - Alternate Pattern 1:
     Select the multiboot_impl_2.bit file in the multiboot/impl_2 folder.
   - Starting Address: 0x01800000 – make sure the value is the same as the value defined for parameter BOOT_ADDRESS_15_TO_0_APPS_IMAGE_1 and BOOT_ADDRESS_31_TO_16_APPS_IMAGE_1 in cmd_list.v.
   - Next Alternate Pattern to Configure: Alternate Pattern 2
     Note: You can select any available option because it no longer affects the next pattern to load once the Dynamic mode is selected.
   - Alternate Pattern 2:
   - Select the multiboot_impl_3.bit file in the multiboot/impl_3 folder.
   - Starting Address: 0x02400000 – make sure the value is the same as the value defined for parameter BOOT_ADDRESS_15_TO_0_APPS_IMAGE_2 and BOOT_ADDRESS_31_TO_16_APPS_IMAGE_2 in cmd_list.v.
   - Next Alternate Pattern to Configure: Primary Pattern
     Note: You can simply select any available option because it no longer affects the next pattern to load once the Dynamic mode is selected.
   - Click **Next**.

© 2026 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.

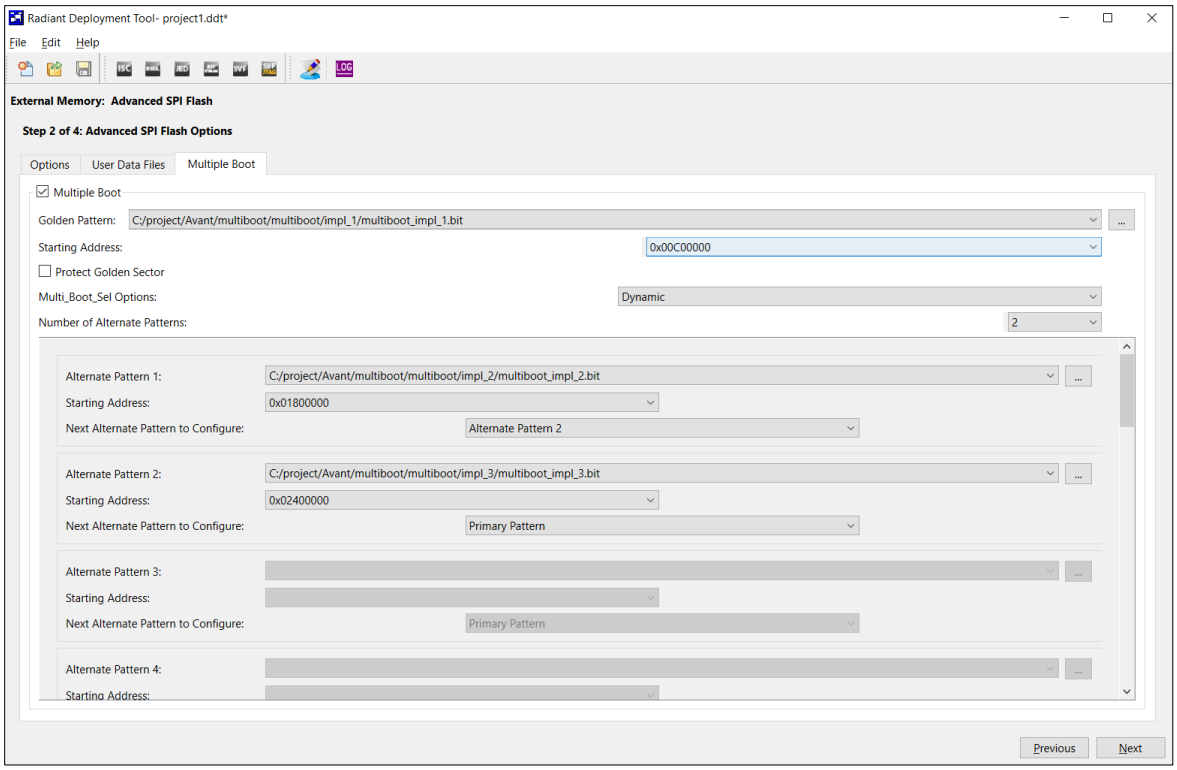

Figure 6.8. Multiple Boot Tab of Step 2 of 4: Advanced SPI Flash Option

d. In Step 3 of 4: Select Output File(s)

- Browse to the location to store the .mcs file in the Output File1 field and click **Next** (Figure 6.9).

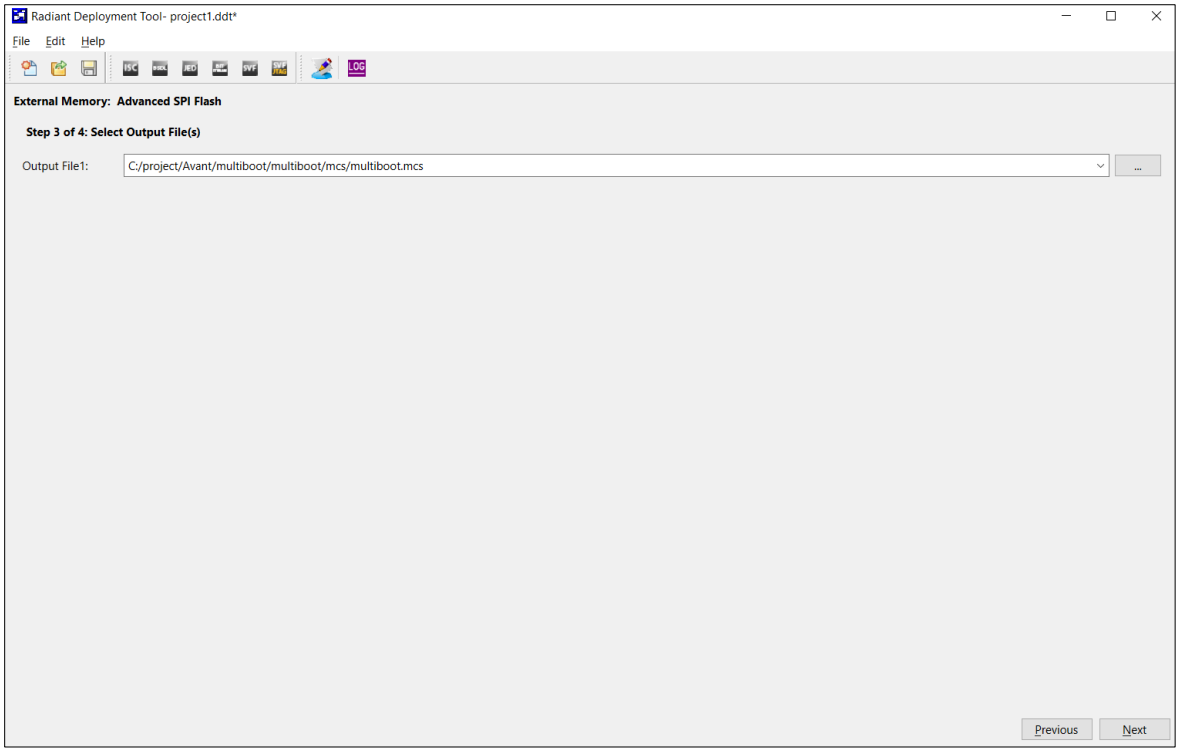

Figure 6.9. Step 3 of 4: Select Output File(s)

© 2026 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.

e. In Step 4 of 4: Generate Deployment
   - Click **Generate** to generate the .mcs file. You can close the window when the Deployment Tool prompts Lattice Deployment Tool has exited successfully (Figure 6.10).

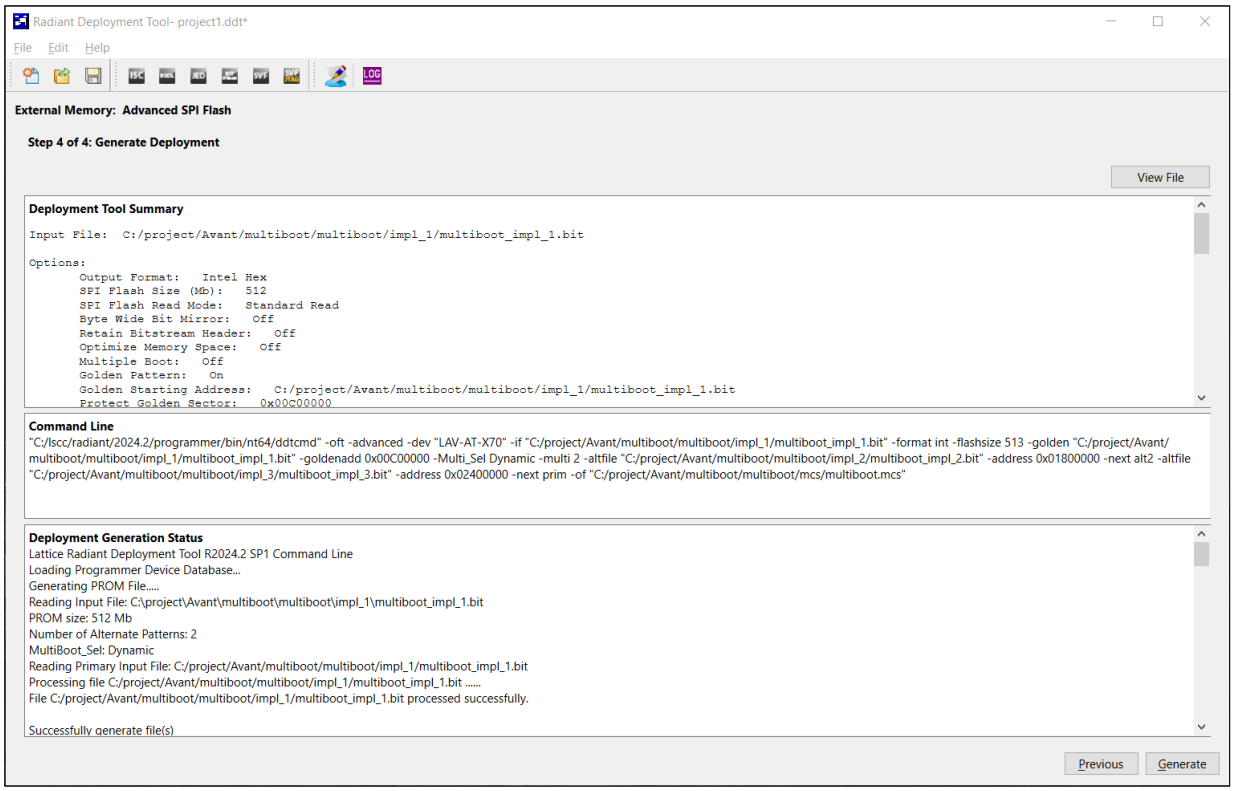

Figure 6.10. Step 4 of 4: Generate Deployment

© 2026 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.

7. Implementing the Reference Design on Board

7.1. Hardware Requirements

- Avant-G/X Versa Board
- USB-A to mini-B Programming Cable

7.2. Programming .mcs into External SPI Flash

1. Launch Radiant Programmer and select **Edit** > **Device Properties**.
2. Select the device properties settings, as shown in Figure 7.1. Click **OK**.
3. In the Cable Settings, select FTUSB-1 for **Port**.
4. Click **Program Device** to start programming the board.

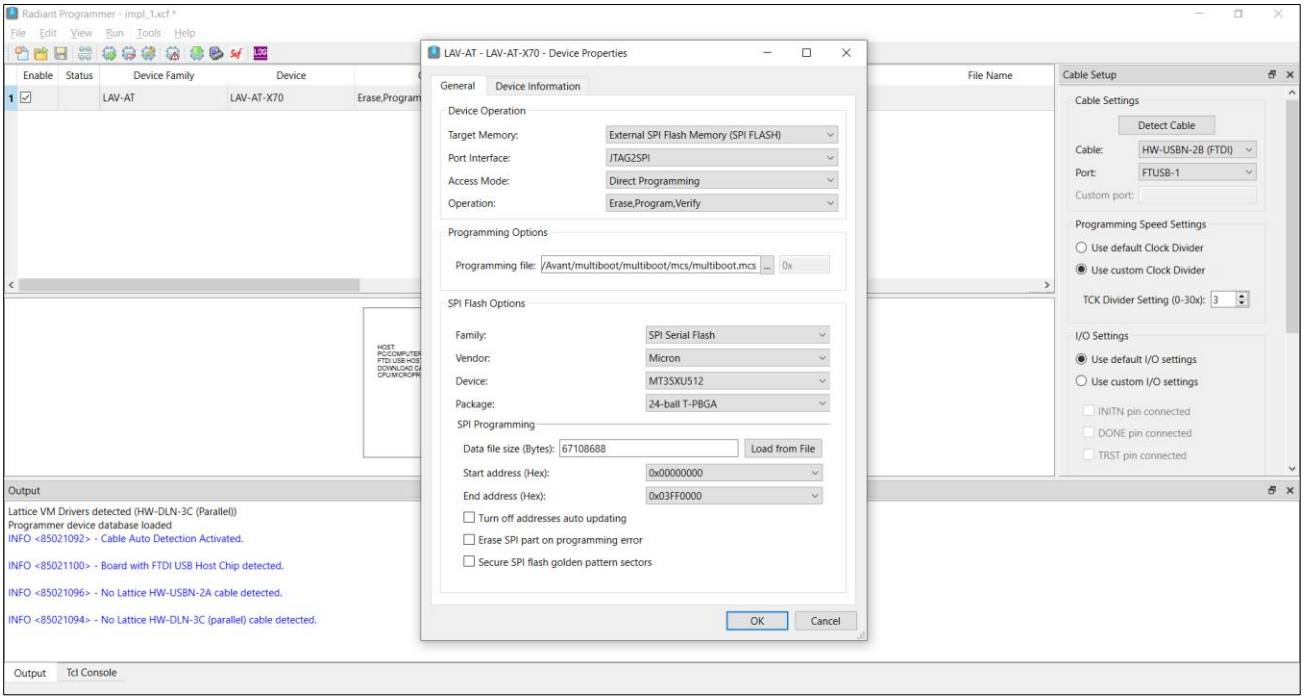

Figure 7.1. Device Properties Window

7.3. Enabling 32-bit MSPI Address and Command Mode

For the multi-boot configuration mode, if the configuration memory is 256 Mb or larger, both the 32-bit SPIM Address and the 32-bit SPIM Command option bits must be set to 1.

To do this, first enable the MSPI_ADDRESS_32BIT and MSPI_COMMAND_32BIT settings in the Global tab of the Lattice Radiant software's Device Constraint Editor. These settings are reflected in the .fea file generated after compiling your project in the Lattice Radiant software.

You can then use this .fea file to program the user OTP settings of the Lattice Avant device. If these bits are not set correctly, the device may fail to boot from Alternate Patterns stored beyond 128 Mb, and the fail-safe mechanism may not function as intended.

© 2026 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.

7.3.1. Setting 32-bit MSPI Address and Command in Non-Volatile Configuration Memory Using Pseudo Programming Mode

This guide allows you to test and ensure that the MSPI configuration from an address beyond 0xFF_FFFF can succeed and therefore the fail-safe mechanism can function properly. The option bits are set in volatile SRAM memory. Setting these option bits is essential to retain the settings in SRAM during reconfiguration. You can perform this test before committing the settings permanently in non-volatile configuration memory.

The following are steps to set the 32-bit MSPI address and 32-bit MSPI commands option bits:

1. Run the Lattice Radiant Programmer.
2. Select the Device Properties settings, as shown in Figure 7.2. Click **OK**.
   - Target Memory: Non Volatile Configuration Memory
   - Port Interface: JTAG
   - Access Mode: Feature Row Pseudo Programming
   - Operation: Update Feature Row

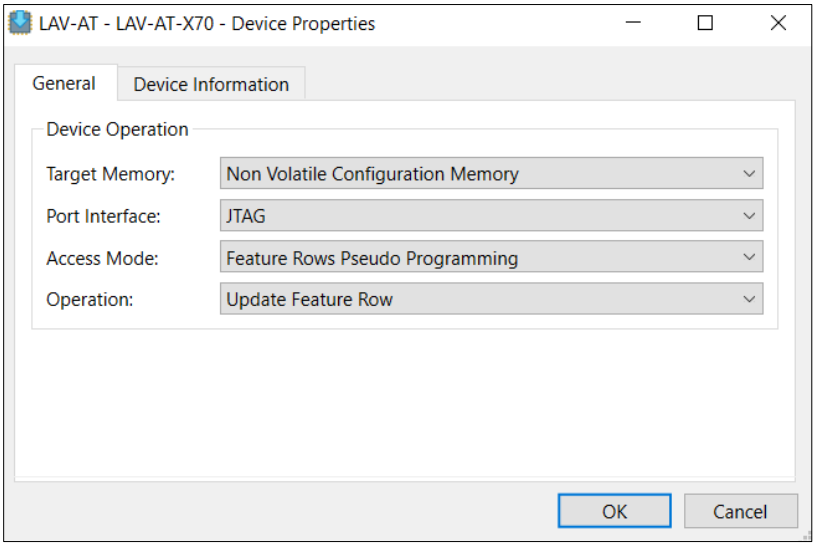

Figure 7.2. Device Properties Window for Feature Row Pseudo Programming

3. In Cable Settings, select FTUSB-1 for **Port**.
4. Click **Run** > **Program Device**. The Feature Row window appears (Figure 7.3).

© 2026 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.

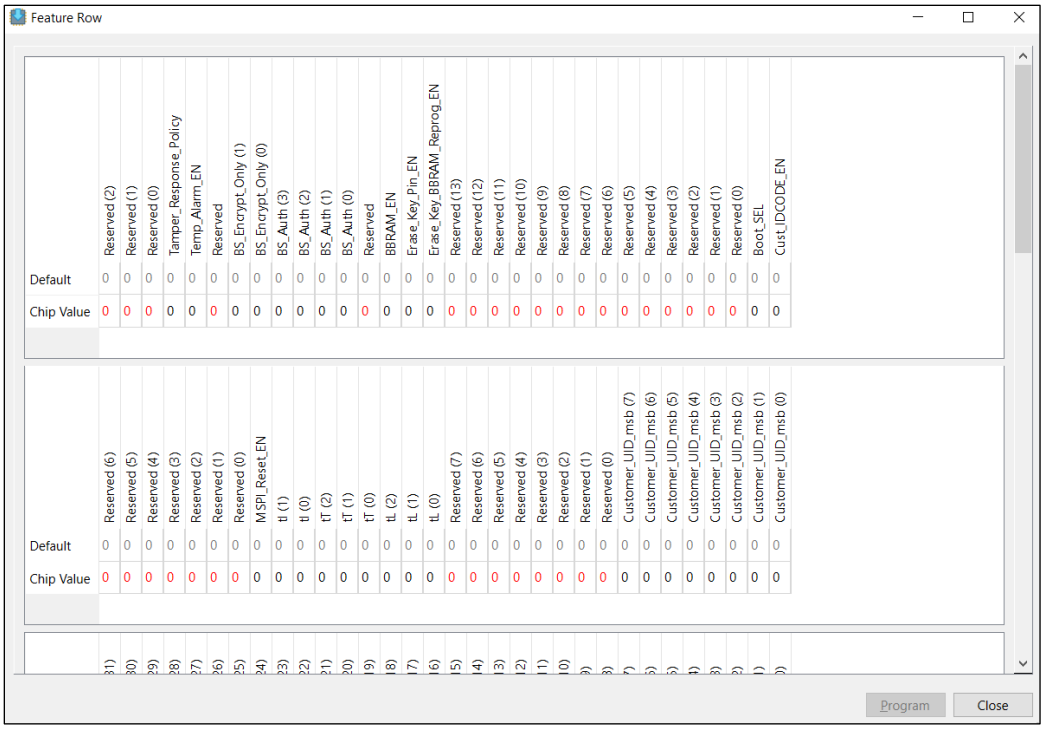

Figure 7.3. Feature Row Window

5. Scroll down to locate the 32-bit SPIM Address and 32-bit SPIM Command bit.
6. Click on the chip values associated with 32-bit SPIM Address and 32-bit SPIM Command to change the values from 0 to 1 (Figure 7.4).

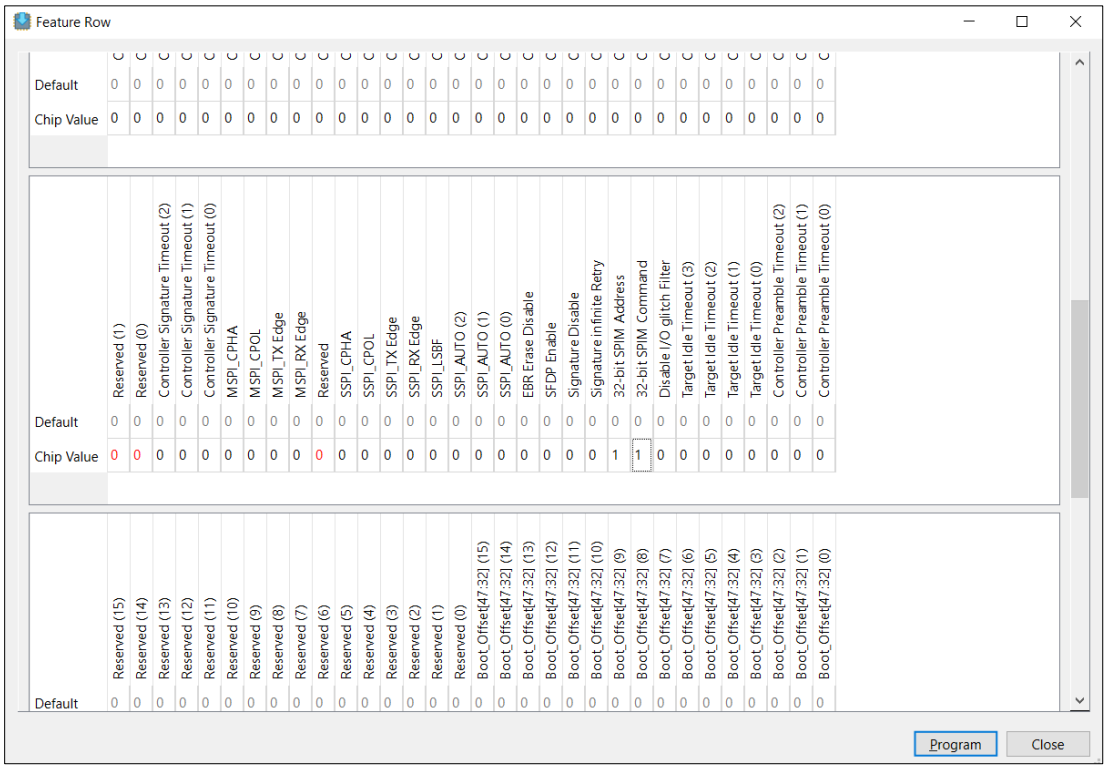

Figure 7.4. Change the Values of 32-bit SPIM Address and 32-bit SPIM Command

© 2026 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.

7. Click **Program**.
8. The Confirm Overwrite Register window appears and Click **Yes** (Figure 7.5).

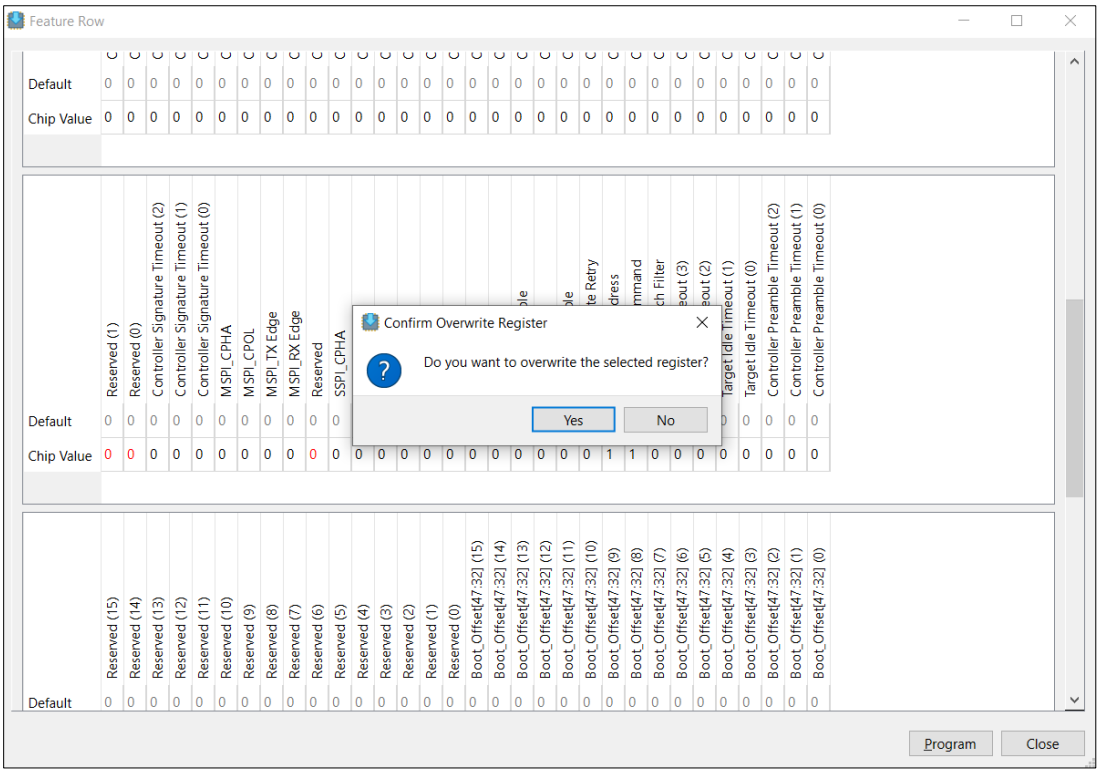

Figure 7.5. Confirm Overwrite Register

9. In the Output Window of Lattice Radiant Programmer, you can see **Operation: successful**, indicating the Feature Row Programming is successful.

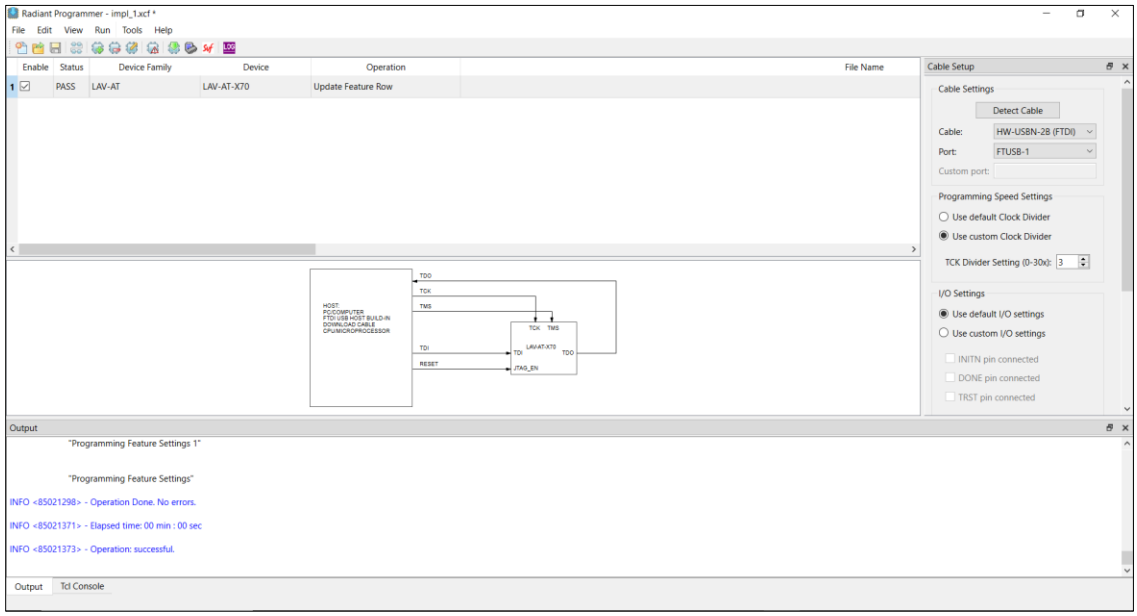

Figure 7.6. Output Window Showing Operation: Successful

© 2026 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.

10. You can repeat steps 4 to 5 to confirm that the 32-bit SPIM Address and 32-bit SPIM Command bits are set to 1.

7.3.2. Setting 32-bit MSPI Address and Command in Non-Volatile Configuration Memory Using Direct Programming Mode

To permanently set the 32-bit SPIM Address and 32-bit SPIM Commands option bits to 1, refer to Appendix C for the steps to perform Feature Row Programming using the .fea file in Avant sysCONFIG User Guide (FPGA-TN-02299).

Note: Feature row bits are one-time programmable (OTP). Each bit can only be modified one time. Once the 32-bit SPIM Address and 32-bit SPIM Commands option bits are set to 1, the FPGA can only work with configuration memory of sizes 256 Mb or larger to perform dual-boot, multi-boot, or ping-pong boot.

7.4. Running the Design

7.4.1. Performing Multi-Boot – without Corrupted Pattern

1. The device is configured with Primary Image after the device is powered on and LED D45 blinks once the device is configured successfully. To confirm this:
   a. Ensure the 32-bit SPIM Address and 32-bit SPIM Command bits are set to 1.
   b. Open the multiboot.rdf project located in the multiboot project directory in the Lattice Radiant software.
   c. Launch and run Reveal Analyzer by double-clicking the **multiboot.rva** file (Figure 7.7).

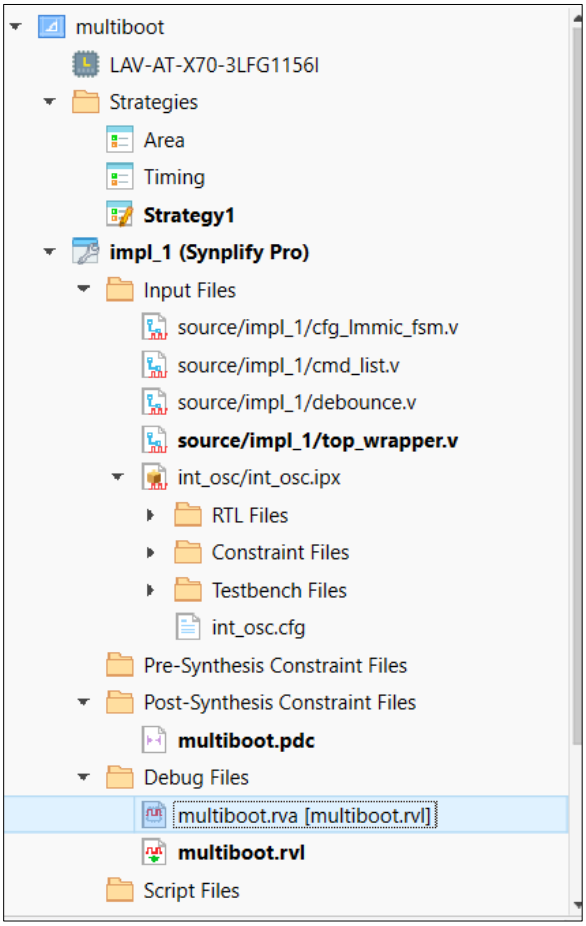

Figure 7.7. multiboot.rva File

© 2026 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.

d. Open the waveform tab and click the button. The apps_image displays CAFE0 indicating that Primary Image is running on the device (Figure 7.8).

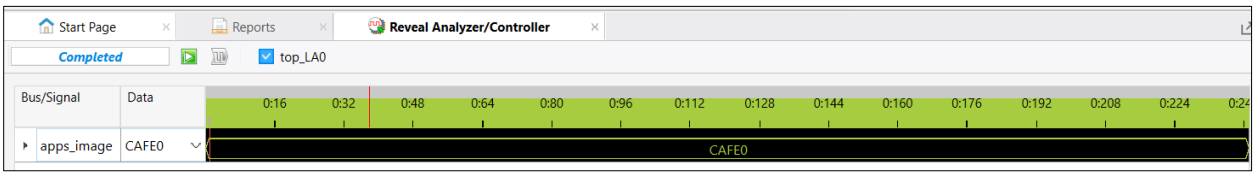

Figure 7.8. apps_image Showing CAFE0

2. Reboot the device with Alternate Pattern 1: Toggle the SW14 push button to reboot or reconfigure the device with Alternate Pattern 1.
3. Run Reveal Analyzer by clicking the button. The apps_image displays CAFE1 and LED D45 and D46 blink, indicating that Alternate Pattern 1 is running on the device (Figure 7.9).

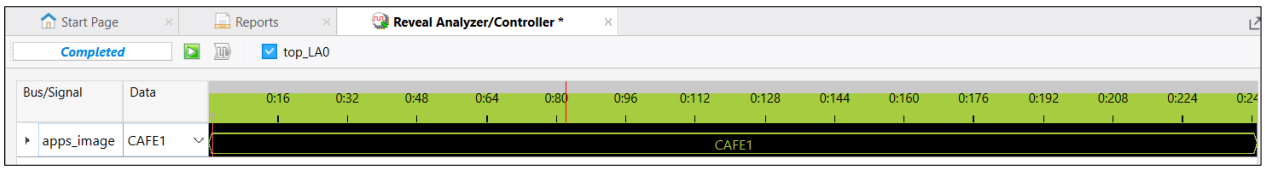

Figure 7.9. apps_image Showing CAFE1

4. Reboot the device with Alternate Pattern 2: Toggle the SW15 push button to reboot or reconfigure the device with Alternate Pattern 2.
5. Run Reveal Analyzer and the apps_image displays CAFE2. LED D45, D46, and D47 blink, indicating that the Alternate Pattern 2 is currently running on the board (Figure 7.10).

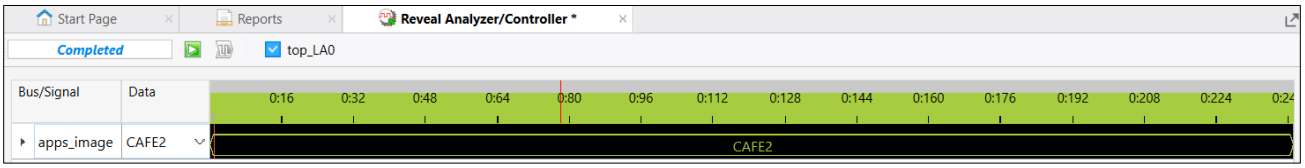

Figure 7.10. apps_image Showing CAFE2

© 2026 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.

7.4.2. Performing Multi-Boot – with Corrupted Pattern

To test the fallback to Golden Pattern function, reprogram Alternate Pattern 1 with a corrupted pattern and then boot the board with the corrupted pattern.

1. Close Reveal Analyzer/Controller and open Radiant Programmer.
2. Reprogram a corrupted image, multiboot/impl_2/multiboot_impl_2_corrupted.bit, at Start Address: 0x01800000 (Figure 7.11).

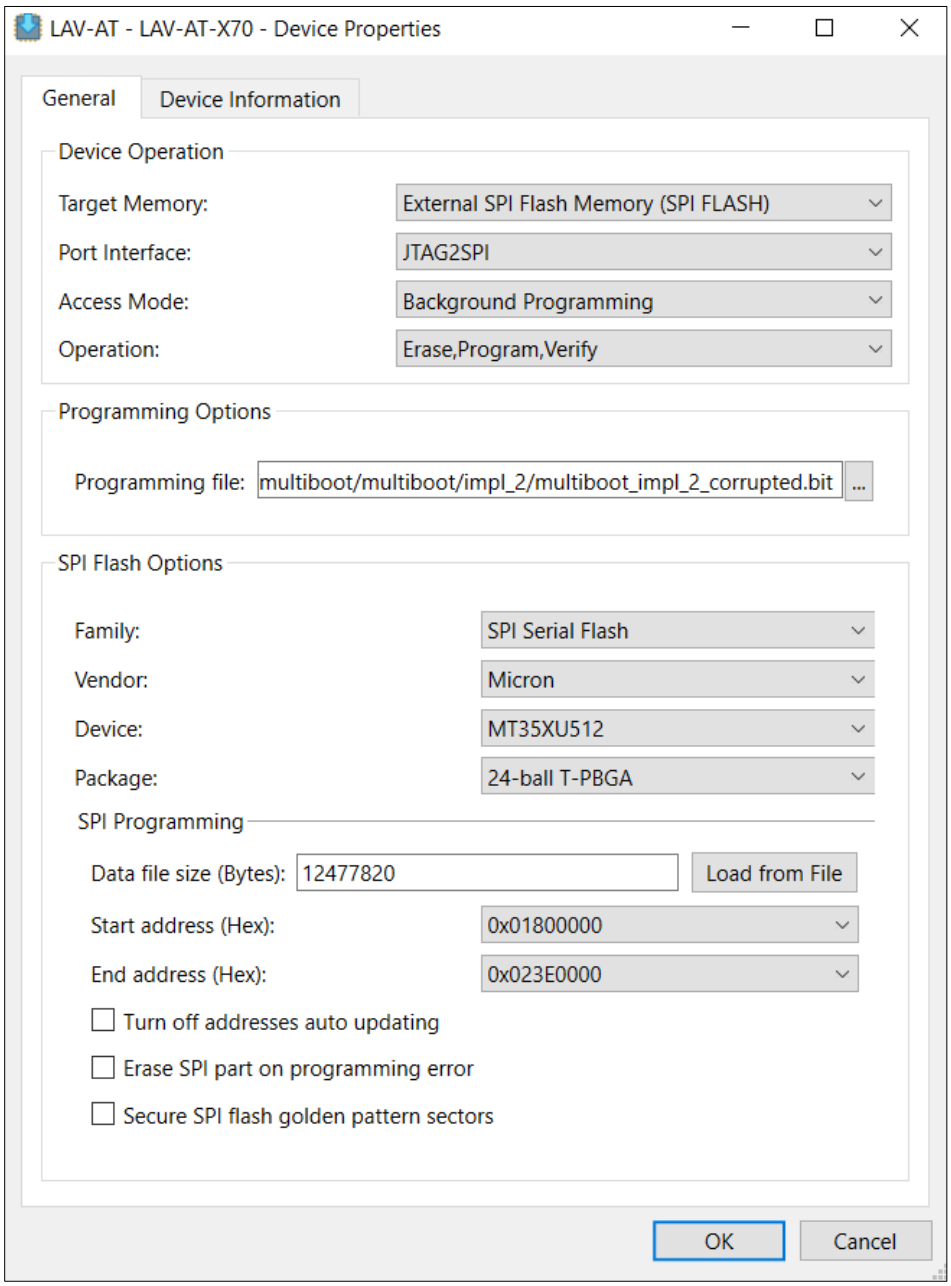

Figure 7.11. Reprogram Corrupted Image

a. After SPI flash programming is completed, toggle the SW14 push button to reboot or reconfigure the device with Alternate Pattern 1.

b. Run Reveal Analyzer and now the apps_image displays CAFE0 (Figure 7.12). LED D45 blinks, indicating that the device fails to boot from the corrupted Alternate Pattern 1 and falls back to Golden Pattern.

© 2026 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.

Note: You are seeing the same output as before because the Golden Pattern is set to be the same as the Primary Pattern.

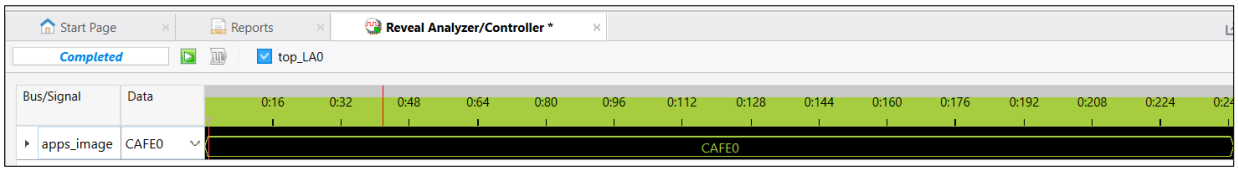

Figure 7.12. apps_image Showing CAFE0

© 2026 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.

References

- Lattice Avant-E Devices web page
- Lattice Avant-G Devices web page
- Lattice Avant-X Devices web page
- Lattice Avant Platform web page
- Lattice Avant Multi-Boot User Guide (FPGA-TN-02314)
- Avant sysCONFIG User Guide (FPGA-TN-02299)
- Lattice Avant OSC Module User Guide (FPGA-IPUG-02184)
- Lattice Radiant Software Help
- Lattice Insights for Lattice Semiconductor training courses and learning plans

© 2026 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.

Technical Support Assistance

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

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

© 2026 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.

Revision History

Revision 1.0, April 2026

Section	Change Summary
All	Updated the document version from 0.80 to 1.0.

Revision 0.80, August 2025

Section	Change Summary
All	Preliminary release.

© 2026 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.

www.latticesemi.com