Lattice Avant Multi-Boot User Guide

Preliminary Technical Note

FPGA-TN-02314-0.86

September 2025

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.

Contents

Figures

Tables

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviation	Definition
CFG	Configuration
CRC	Cyclic Redundancy Check
EBR	Embedded Block RAM
ECDSA	Elliptic Curve Digital Signature Algorithm
FPGA	Field Programmable Gate Array
FSM	Finite State Machine
GUI	Graphic User Interface
ID	Identification
JTAG	Joint Test Action Group
LMMI	Lattice Memory Mapped Interface
MSPI	Controller Serial Peripheral Interface
OSC	Oscillator
PC	Personal Computer
RAM	Random Access Memory
RSA	Rivest-Shamir-Adleman (cryptosystem)
SFDP	Serial Flash Discoverable Parameter
SPI	Serial Peripheral Interface
SRAM	Static Random-Access Memory
T-PBGA	Thin Plastic Ball Grid Array
xSPI	Expanded Serial Peripheral Interface

1. Introduction

The Lattice Avant™ platform supports various booting options for loading configuration SRAM from non-volatile memory. These booting options provide configuration flexibility and facilitate fail-safe configuration. Avant devices require an external memory for storage of the configuration bitstreams.

Multiple configuration boot modes mitigate risk during the field upgrade process and allow flexibility of executing different patterns. Field upgrade disruptions might occur due to power disruption, communication interruption, or bitstream pattern corruption. The Avant platform supports the following boot modes:

- Single bitstream boot mode
- Dua-boot mode
- Ping-pong boot mode
- Multi-boot mode

The Avant platform combines multiple bitstream patterns into a single boot image stored in a single external SPI storage device. This solution decreases cost, reduces board space, and simplifies field upgrades.

2. Definition of Terms

Table 2.1 lists the terms used in this document to describe common functions, features, or concepts.

Table 2.1. Definition of Terms

Term	Definition
Alternative Boot Pattern	In multi-boot mode, after the FPGA device has been configured, this pattern is loaded after PROGRAMN pin pulsing or REFRESH command execution. Up to 16 alternative boot patterns are possible. An alternative boot pattern is also referred to as an alternate pattern in this document.
Binary Hex Data File (.bin File)	The data image of the Hex data file in binary format. All Hex data files are converted into this format prior to consumption.
Bitstream Data File (.bit File)	The configuration data file, for a single FPGA device, in the format that can be loaded directly into the FPGA device to configure the SRAM cells. The file is expressed in binary format.
Configuration	A change in the state of the SRAM memory cells.
Dual-Boot	This feature allows the FPGA device to support two configuration images that reside in an SPI storage device. Whenever loading failure occurs with the primary image, the FPGA device searches for and loads the secondary image. Both images come from non-volatile SPI memory.
Flash Lock	The feature provides protection to the flash memory against accidental erase or corruption. Most SPI flash devices support soft lock. Lock choices include: • Whole device • Bottom half • Bottom quarter • Last sector Details can be found in the SPI flash device data sheet.
Golden Pattern	The guaranteed good pattern loaded into the FPGA device when booting failure occurs. Only one Golden pattern is allowed.
Hex Data File, Hex File (.exo, .mcs, .xtek Files)	A data record file commonly in a format such as Intel Hex, Motorola Hex, or Extended Tektronix Hex. This file is also known as an addressed record file. It is used for programming configuration memory or flash devices.
MCLKP	On the controller SPI port, MCLKP is the output clock signal used to drive an external memory device to sequentially load configuration data for the FPGA.
Multi-Boot	The FPGA device determines and triggers the loading of the next pattern after a prior successful configuration. Multiple patterns (that is, two patterns or more) are available for the FPGA device to choose to load on demand. All patterns are stored in non-volatile SPI memory.
Ping-Pong Boot	This feature allows the FPGA device to utilize the jump table to select a pattern for booting without changing the location of the pattern in the SPI storage device.
Primary Pattern	At power up, the FPGA device loads this pattern first. Only one Primary pattern is allowed.
Programming	The process used to alter the contents of the external configuration memory.
Refresh	The process of triggering a configuration data load operation. It is activated by PROGRAMN pin pulsing or REFRESH command execution (which emulates PROGRAMN pin pulsing).
Sector (Block)	The smallest number of bytes of flash memory that can be erased at the same time by the erase command.

3. Resources

Avant devices are SRAM-based FPGAs. The SRAM configuration memory must be loaded from an external non-volatile memory that can store all the configuration data. The size of the configuration data varies. It is dependent on the amount of logic available in the FPGA and the number of pre-initialized Embedded Block RAM (EBR) components. A design using the largest Avant device, with every EBR pre-initialized with unique data values and generated without compression enabled, requires the largest amount of storage.

Table 3.1. Bitstream Size versus Recommended SPI Storage Size (Single Bitstream Boot Mode)

Device	Scenario	Bitstream Size (Mb)[1]	Recommended SPI Storage Device Size (Mb)
LAV-AT-E/G/X30	No EBR	44	64
	Maximum EBR	64.1	128
LAV-AT-E/G/X50	No EBR	99.8	128
	Maximum EBR	131.4	256
LAV-AT-E/G/X70	No EBR	99.8	128
	Maximum EBR	149.4	256

Note:

1. Both unencrypted and encrypted bitstreams are the same size. Bitstream compression ratio varies depending on the bitstream so only uncompressed bitstream sizes are shown.

Table 3.2. Bitstream Size versus Recommended SPI Storage Size (Dual-Boot Mode, Ping-Pong Boot Mode)

Device	Scenario	Bitstream Size (Mb)[1]		Recommended SPI Storage Device Size (Mb)
		1 Bitstream Pattern	2 Bitstream Patterns	
LAV-AT-E/G/X30	No EBR	44	88	128
	Maximum EBR	64.1	128.2	256
LAV-AT-E/G/X50	No EBR	99.8	199.6	256
	Maximum EBR	131.4	262.8	512
LAV-AT-E/G/X70	No EBR	99.8	199.6	256
	Maximum EBR	149.4	298.8	512

Note:

1. Both unencrypted and encrypted bitstreams are the same size. Bitstream compression ratio varies depending on the bitstream so only uncompressed bitstream sizes are shown.

Table 3.3. Bitstream Size versus Recommended SPI Storage Size (Multi-Boot Mode)

Device	Scenario	Bitstream Size (Mb)[1]					Recommended SPI Storage Device Size (Mb)			
		1 Bitstream Pattern	3 Bitstream Patterns	4 Bitstream Patterns	5 Bitstream Patterns	6 Bitstream Patterns	3 Bitstream Patterns	4 Bitstream Patterns	5 Bitstream Patterns	6 Bitstream Patterns
LAV-AT-E/G/X30	No EBR	44	132	176	220	264	256	256	256	512
	Maximum EBR	64.1	192.3	256.4	320.5	384.6	256	512	512	512
LAV-AT-E/G/X50	No EBR	99.8	299.4	399.2	499	598.8	512	512	512	1024
	Maximum EBR	131.4	394.2	525.6	657	788.4	512	1024	1024	1024
LAV-AT-E/G/X70	No EBR	99.8	299.4	399.2	499	598.8	512	512	512	1024
	Maximum EBR	149.4	448.2	597.6	747	896.4	512	1024	1024	1024

Note:

1. Both unencrypted and encrypted bitstreams are the same size. Bitstream compression ratio varies depending on the bitstream so only uncompressed bitstream sizes are shown.

4. Dual-Boot Mode

The Avant device dual-boot mode supports booting from two configuration patterns that reside in an external SPI storage device. These patterns are designated the primary pattern and golden pattern. When an Avant device boots up, it attempts to boot from the primary pattern. If the primary pattern fails to load, the device attempts to boot from the golden pattern. Figure 4.1 shows the dual-boot flow diagram.

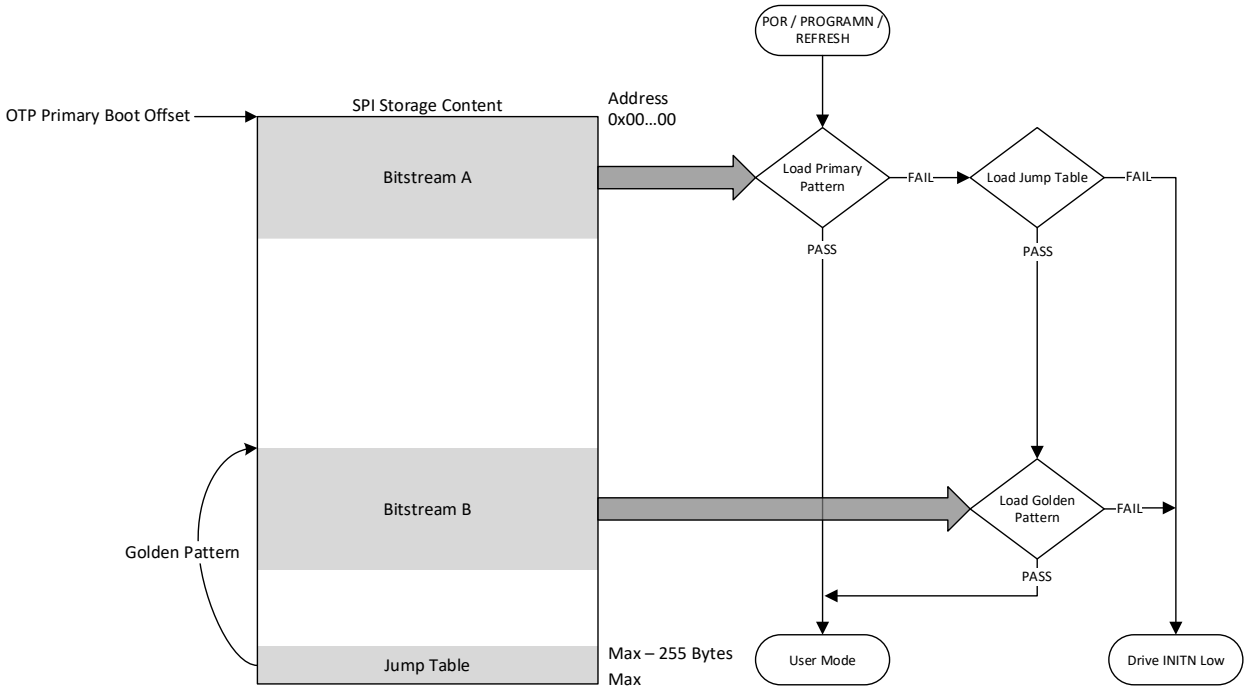


Figure 4.1. Dual-Boot Flow Diagram

4.1. Dual-Boot Flow

The dual-boot mode flow is triggered either at power up, PROGRAMN pin pulsing, or REFRESH command execution.

In dual-boot mode, the primary pattern might fail to load for one of the following reasons:

- Time-out check — Search for a 32-bit preamble code (0xFFFFBxB3) in the primary pattern, as part of the configuration process, fails.
- Device ID check – ID check at the beginning of the bitstream fails.
- Data corruption check — CRC check detect data corruption in the bitstream.
- An illegal command is encountered.

Table 4.1 shows the time-out period when loading the primary pattern stored in the external SPI storage device.

Table 4.1. Control Register 1 [2:0] – MSPI Preamble Timer Count

CR1[2:0]	Timer Value
0	200 ms (default setting)
1	100 ms
2	50 ms
3	40 ms
4	20 ms
5	1 ms
6	500 µs
7	100 µs

If the primary pattern fails to load, the Avant device drives the INITN pin low to indicate an error and resets the configuration engine. After clearing the configuration RAM, the device drives the INITN pin high and reads the JUMP command in the jump table that directs it to load the golden pattern in the SPI storage device.

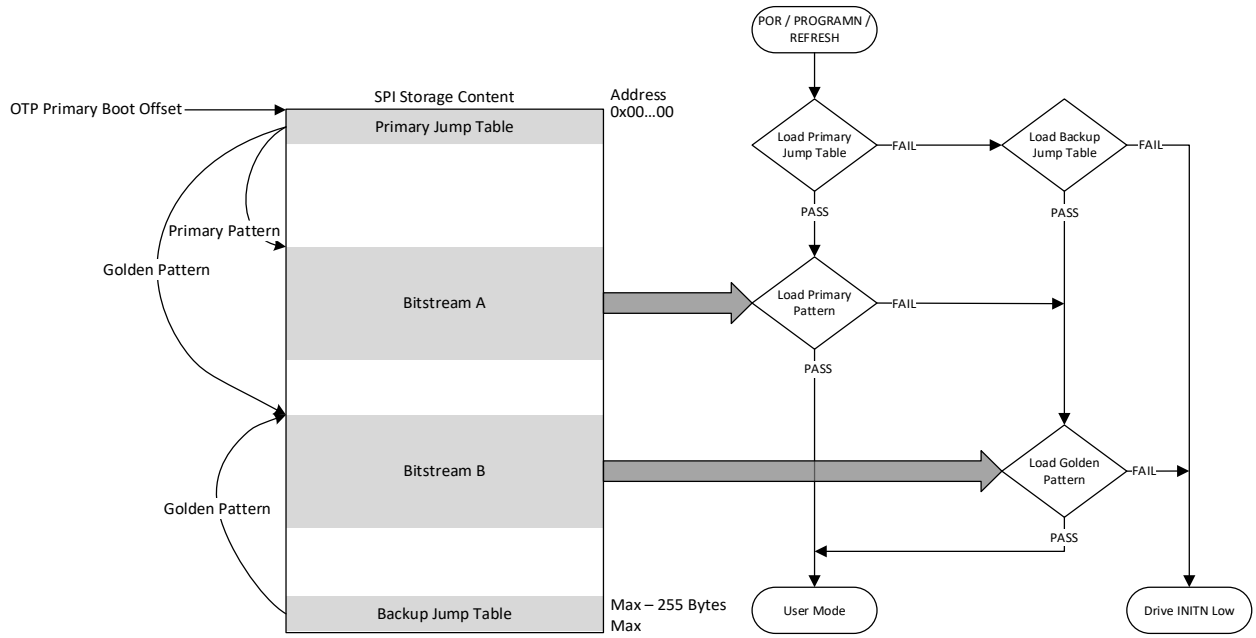
Note: The jump table is located at the last 256 bytes of the SPI storage device. When the configuration engine attempts to read the jump table, it issues a read offset address at 0xFFFF00 for 24-bit SPI flash, 0xFFFF_FF00 for 32-bit SPI flash, or 0xFFFF_FFFF_FF00 for 45-bit xSPI flash. The SPI flash truncates the most significant bit or bits of the address depending on the density of the SPI flash. For example, a 32-Mb SPI flash receives the read offset address as 0x3FFF00 even though the configuration engine sends 0xFFFF00.

Note: By default, the 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row are set to 0. The MSPI controller operates in 3-byte command and addressing mode. This enables the device to support dual-boot, multi-boot, or ping-pong boot mode with configuration memory sizes up to 128 Mb. If your configuration memory size is 256 Mb or larger, the 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row must be set to 1.

The Avant device performs the same checks for the golden pattern. If the golden pattern is also corrupted, configuration fails. The Avant device stops driving the SPI clock and drives the INITN pin low.

5. Ping-Pong Boot Mode

The Avant device ping-pong boot mode supports booting from one of the two configuration bitstream patterns that resides in an external SPI storage device. These patterns are designated the primary bitstream and secondary bitstream. The Avant device boots from the bitstream pattern based on the assignment in the jump table. The jump table allows the device to boot from either bitstream without changing the physical location of the two bitstreams within the external SPI storage device. Only the jump table needs to be updated to change the boot bitstream pattern. When one bitstream is selected as the primary bitstream pattern, by default, the secondary bitstream becomes the golden bitstream pattern. The external SPI storage device can also store a backup jump table, in case the primary jump table becomes corrupted. Figure 5.1 shows the ping-pong boot flow diagram.



Notes:
1. The primary jump table points to the addresses of the primary and golden patterns.
2. The backup jump table points to the golden pattern.
3. To change the designation of which bitstream is primary and which is golden, only the jump tables need to be updated.

Figure 5.1. Ping-Pong Boot Flow Diagram

5.1. Ping-Pong Boot Flow

The ping-pong boot flow is triggered at power up, PROGRAMN pin pulsing, or REFRESH command execution. When a jump command in the primary jump table is executed, the device stops the SPI clock, drives the INITN pin low, resets the configuration engine, and clears the configuration RAM. After the configuration RAM is cleared, the device drives the INITN pin high, restarts the SPI clock, and reads the primary pattern from the SPI storage device as addressed by the jump table.

In ping-pong boot mode, the bitstream pattern selected might fail to load for one of the following reasons:

- Time-out check — Search for a 32-bit preamble code (0xFFFFBxB3) in the bitstream pattern, as part of the configuration protocol, fails. Refer to Table 4.1 for the time-out values.
- Device ID check – ID check at the beginning of the bitstream fails.
- Data corruption check — CRC checks detect data corruption in the bitstream.
- An illegal command is encountered.

If the bitstream fails to load, the Avant device drives the INITN pin low to indicate an error and resets the configuration engine. After clearing the configuration RAM, the device drives the INITN pin high and reads the address of the golden pattern in the backup jump table. If the golden pattern configuration fails, the Avant device stops driving the SPI clock and drives the INITN pin low.

If the primary jump table is corrupted, the device loads the backup jump table, which then loads the golden pattern.

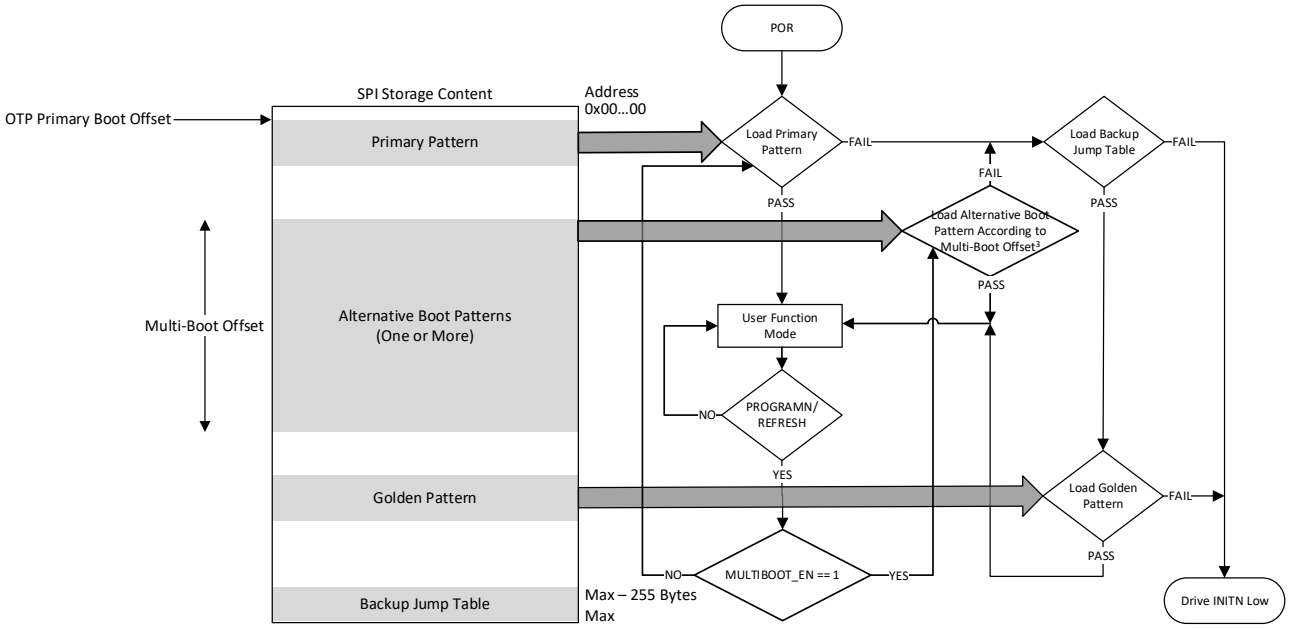
Note: The backup jump table is located at the last 256 bytes of the SPI storage device. When the configuration engine attempts to read the backup jump table, it issues a read offset address at 0xFFFF00 for 24-bit SPI flash, 0xFFFF_FF00 for 32-bit SPI flash, or 0xFFFF_FFFF_FF00 for 45-bit xSPI flash. The SPI flash truncates the most significant bit or bits of the address depending on the density of the SPI flash. For example, the 32-Mb SPI flash receives the read offset address as 0x3FFF00 even though the configuration engine sends 0xFFFF00.

Note: By default, the 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row are set to 0. The MSPI controller operates in 3-byte command and addressing mode. This enables the device to support dual-boot, multi-boot, or ping-pong boot mode with configuration memory sizes up to 128 Mb. If your configuration memory size is 256 Mb or larger, the 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row must be set to 1.

If the backup jump table is corrupted or the golden pattern configuration fails, the Avant device stops driving the SPI clock and drives the INITN pin low.

6. Multi-Boot Mode

The Avant device multi-boot mode supports booting from at least three and up to 18 configuration bitstream patterns that reside in an external SPI storage device. These patterns are designated the primary pattern, golden pattern, and alternative boot patterns. When an Avant device boots up, it attempts to boot from the primary pattern. If the primary pattern fails to load, the device attempts to boot from the golden pattern. After successful configuration, the MULTI_BOOT_SEL attribute determines the subsequent loading of alternative boot patterns. Figure 6.1 shows the multi-boot flow diagram.



Notes:
1. The default primary boot offset is at address 0.
2. When MULTI_BOOT_SEL == STATIC, the multi-boot offset uses the value stored in the Multiboot Offset register. When MULTI_BOOT_SEL == DYNAMIC, the multi-boot offset uses the multi-boot address register programmed through the CONFIG_LMMI interface.
3. Load the primary pattern if the multi-boot offset points to the start address of the primary pattern in the SPI storage device.

Figure 6.1. Multi-Boot Flow Diagram

6.1. Multi-Boot Flow

If MULTI_BOOT_SEL == STATIC, after the primary pattern loads, reprogramming of the bitstream can be done by performing PROGRAMN pin pulsing or REFRESH command execution, which then triggers the loading of alternative boot pattern 1. Subsequent PROGRAMN pin pulsing or REFRESH command execution events load the next pattern defined in the multi-boot configuration. The bitstream pattern sequence, target address of the golden pattern, and target addresses of the alternative boot patterns are defined during the multi-boot configuration process in the Lattice Radiant™ Deployment Tool.

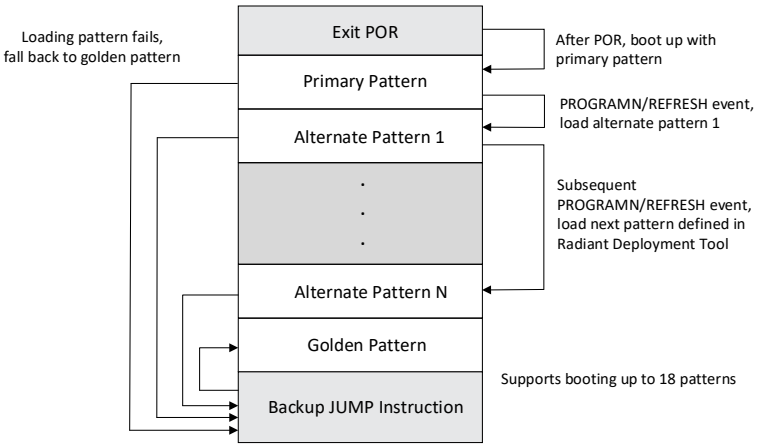


Figure 6.2. Static Multi-Boot Mode Flow

If MULTI_BOOT_SEL == DYNAMIC, after the primary pattern loads, you can switch between all the different patterns stored in the external storage device freely.

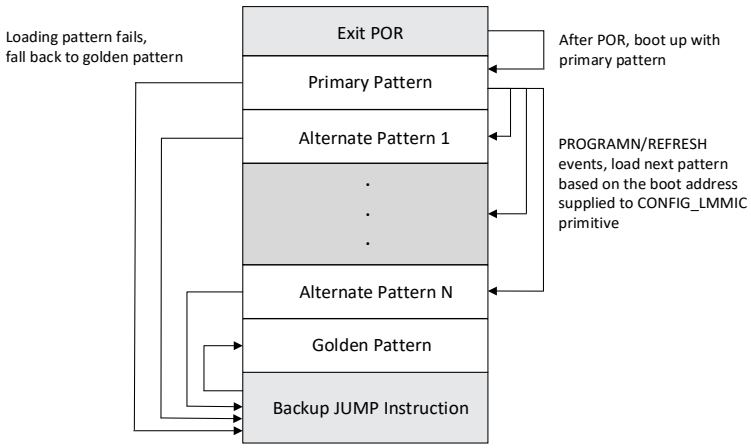


Figure 6.3. Dynamic Multi-Boot Mode Flow

If the primary pattern or alternative boot pattern fails to load, the device drives the INITN pin low to indicate an error and resets the configuration engine. After clearing the configuration RAM, the device drives the INITN pin high and reads the JUMP command in the backup jump table that directs it to load the golden pattern in the SPI storage device. If the golden pattern configuration fails, the Avant device stops driving the SPI clock and drives the INITN pin low. Once golden pattern configuration fails, you must reprogram the external SPI flash through the JTAG interface (using the JTAG-to-Controller SPI Bridge) or target SPI port (using the Target SPI-to-Controller SPI Bridge).

Note: The backup jump table is located at the last 256 bytes of the SPI storage device. When the configuration engine attempts to read the backup jump table, it issues a read offset address at 0xFFFF00 for 24-bit SPI flash, 0xFFFF_FF00 for 32-bit SPI flash, or 0xFFFF_FFFF_FF00 for 45-bit xSPI flash. The SPI flash truncates the most significant bit or bits of the address depending on the density of the SPI flash. For example, the 32-Mb SPI flash receives the read offset address as 0x3FFF00 even though the configuration engine sends 0xFFFF00.

Note: By default, the 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row are set to 0. The MSPI controller operates in 3-byte command and addressing mode. This enables the device to support dual-boot, multi-boot, or ping-pong boot mode with configuration memory sizes up to 128 Mb. If your configuration memory size is 256 Mb or larger, the 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row must be set to 1.

The software selection options for the multi-boot mode can be set through the Global tab of the Device Constraint Editor in the Lattice Radiant software.

- MULTI_BOOT_MODE: Enables multi-boot mode.
- MULTI_BOOT_SEL: Selects the booting address for the multiple boot event when MULTI_BOOT_MODE is set to ENABLE.
  - STATIC (default): Use the value stored in the MultiBoot Offset register as the booting address. This bitstream pattern booting address value can be preset through the Global tab of the Device Constraint Editor. Set these values for the bitstream patterns of your boot sequence before you use the Lattice Radiant Deployment Tool to create the multi-boot hex file. Refer to the MultiBoot Offset section in the Lattice Avant sysCONFIG User Guide (FPGA-TN-02299) for more information about the MultiBoot Offset register.
  - DYNAMIC: Use the multi-boot address register programmed through the CONFIG_LMMI interface.
- MultiBoot Offset: A 48-bit register stores the boot address for the next pattern to load from the external SPI flash. When MULTI_BOOT_SEL is set to STATIC, the device uses the value stored in this register as the booting address for the next pattern.

  Note: If you use the STATIC setting for the MULTI_BOOT_SEL option, whenever updating a particular alternate pattern in the external SPI flash, ensure the corresponding bitstream pattern booting address value is updated accordingly, if necessary, through the Device Constraint Editor. A booting address value update is necessary when the bitstream pattern location changes.

6.2. Implementation of Static or Dynamic Multi-Boot Mode

To implement static or dynamic multi-boot mode, the CONFIG_LMMIC primitive as shown in Figure 6.4 must be incorporated into your design.

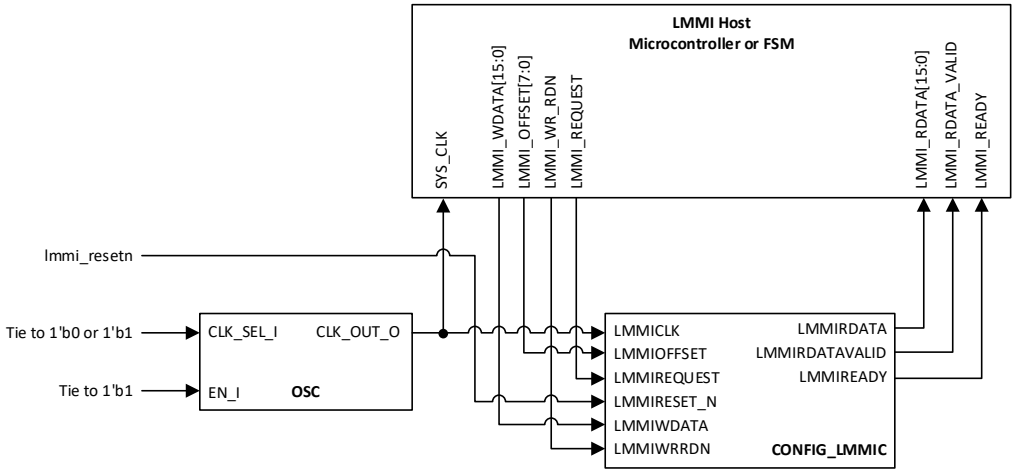


Figure 6.4. CONFIG_LMMI Primitive for Multi-Boot Implementation

The following are the blocks or modules required for this implementation:

- OSC: The oscillator module generates the clock source for the LMMI Host and CONFIG_LMMIC primitive. Refer to the Lattice Avant OSC Module User Guide (FPGA-IPUG-02184) for more information.
- CONFIG_LMMIC: The CONFIG_LMMIC primitive adds the Lattice memory mapped interface (LMMI) to the configuration block. Refer to the CONFIG_LMMIC page in the Lattice Radiant Software Help for more information.
- LMMI Host: The LMMI Host 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 sequence to load the pattern for the static multi-boot mode or dynamic multi-boot mode.

For a multi-boot reference design, refer to the Lattice Avant Multi-Boot Reference Design (FPGA-RD-02321).

6.2.1. Static Multi-Boot Mode Host Controller Sequence

The following is the controller sequence for loading the pattern in static multi-boot mode, with the flow as shown in Figure 6.5:

1. LMMI_CFG_PORT_REQUEST[15:0] and LMMI_CFG_PORT_REQUEST[31:16]: Executes the port request command to enable exclusive LMMI access to the configuration engine. 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, thereby requiring power cycling of the device to recover the LMMI port.
2. LMMI_CFG_PORT_STATUS[15:0]: Reads the port status for the CONFIG_LMMI interface and proceeds to the next state once exclusive access is granted.
3. REFRESH: Equivalent to pulsing the PROGRAMN pin. Once this command is executed, the device starts loading the desired alternate pattern from the external SPI flash according to the configuration sequence defined in the multi-boot configuration process in the Lattice Radiant Deployment Tool. If loading of the pattern fails, the device falls back to loading the golden pattern.

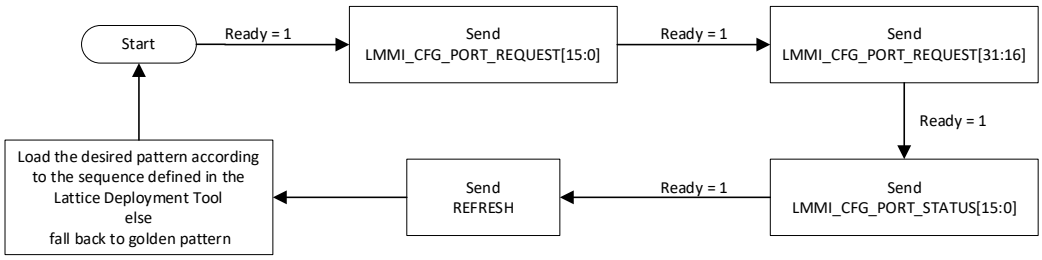


Figure 6.5: Host Controller Sequence for Static Multi-Boot Mode

For more information on configuration access through LMMI and target configuration commands, refer to the LMMI CFG Offset List and Target Configuration Commands tables, respectively, in the Lattice Avant sysCONFIG User Guide (FPGA-TN- 02299).

6.2.2. Dynamic Multi-Boot Mode Host Controller Sequence

The following is the controller sequence for loading the pattern in dynamic multi-boot mode, with the flow as shown in Figure 6.6:

1. LMMI_CFG_PORT_REQUEST[15:0] and LMMI_CFG_PORT_REQUEST[31:16]: Executes the port request command to enable exclusive LMMI access to the configuration engine. 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, thereby requiring power cycling of the device to recover the LMMI port.
2. LMMI_CFG_PORT_STATUS[15:0]: Reads the port status for the CONFIG_LMMI interface and proceeds to the next state once exclusive access is granted.
3. LMMI_CFG_MSPI_MULTIBOOT_ADDR [15:0] and LMMI_CFG_MSPI_MULTIBOOT_ADDR [31:16]: Sends the start address through the CONFIG_LMMIC primitive. The start address specifies the location in the external SPI flash where the desired bitstream is stored for booting and configuring the device.

4. REFRESH: Equivalent to pulsing the PROGRAMN pin. Once this command is executed, the device starts loading the desired alternate pattern from the external SPI flash according to the boot address sent in the previous state. If loading of the pattern fails, the device falls back to loading the golden pattern.

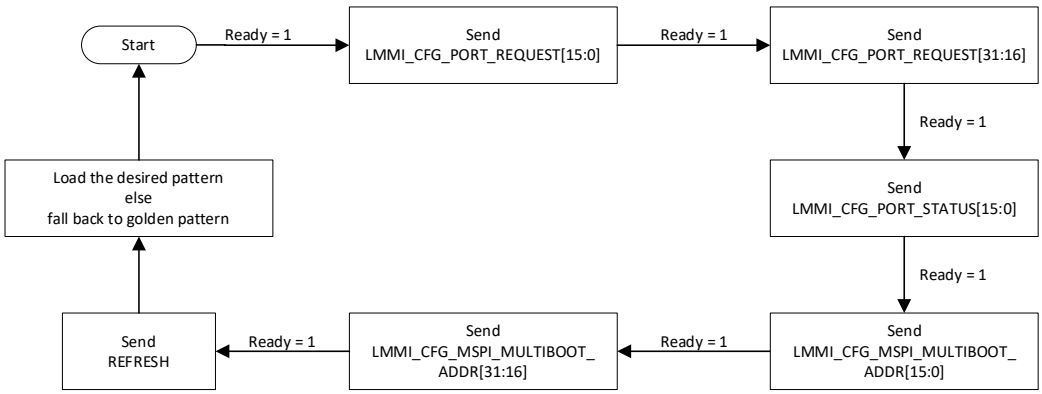


Figure 6.6. Host Controller Sequence for Dynamic Multi-Boot Mode

For more information on configuration access through LMMI and target configuration commands, refer to the LMMI CFG Offset List and Target Configuration Commands tables, respectively, in the Lattice Avant sysCONFIG User Guide (FPGA-TN- 02299).

7. Creating an SPI Storage Device Hex File

The Lattice Radiant software provides a turnkey solution to manage the low-level implementation details of the dual-boot, ping-pong boot, and multi-boot features on Lattice Avant family devices. The Lattice Deployment Tool, which is part of Lattice Radiant software, merges the different patterns and the JUMP command into one SPI storage device hex file. The SPI storage device hex file can later be programmed into an external SPI storage device using the Radiant Programmer or a third-party programmer.

7.1. Using Radiant Deployment Tool to Create a Dual-Boot Hex File

The section provides the procedure for generating a dual-boot SPI storage device hex file using the Radiant Deployment Tool.

- Generate the golden and primary bitstream files in the Lattice Radiant software.
  - MCCLK_FREQ (MCLKP frequency) setting of bitstream files should not exceed the external SPI storage device normal/standard read speed.
  - MCCLK_FREQ can be configured using the Global tab of the Device Constraint Editor in the Lattice Radiant software.
- Invoke the Lattice Radiant Deployment Tool by choosing **Tools > Deployment Tool** from the Lattice Radiant Programmer.
- In the Radiant Deployment Tool window, select **External Memory** as the **Function Type** and select **Dual Boot** as the **Output File Type** (Figure 7.1).
- Click **OK**.

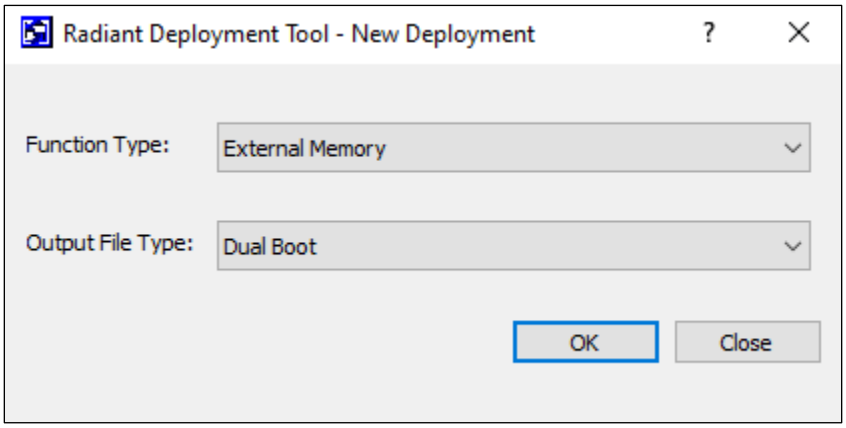


Figure 7.1. Creating New Deployment for Dual Boot File

Step 1 of 4: Select Input File(s) window (Figure 7.2)

- Click on the **…** button in the **File Name** fields to browse and select the two bitstream files to be used to create the SPI storage device hex file.
- The **Device Family** and **Device** fields automatically populate based on the bitstream files selected.
- Click **Next**.

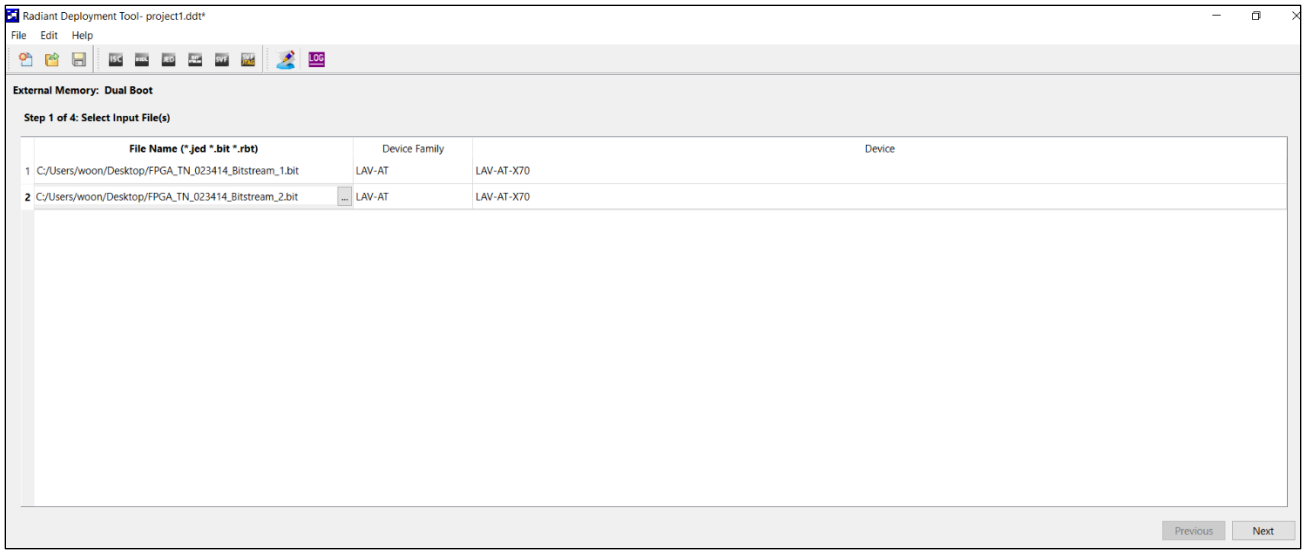


Figure 7.2. Select Input File(s) Window

Step 2 of 4: Dual Boot Options window (Figure 7.3)

- Select the **Output Format** (Intel Hex, Motorola Hex, or Extended Tektronix Hex).
- Select the **SPI Flash Size (Mb)** (4, 8, 16, 32, 64, 128, 256, 512, or 1024 Mb).
- Select **SPI Flash Read Mode** (Standard Read, Fast Read, Dual I/O SPI Flash Read, or Quad I/O SPI Flash Read).
- Single-ended Clock is the only available option for **Clock Mode** field.
- The Radiant Deployment Tool automatically assigns the bitstream files selected in Step 1 to be used for the golden pattern and primary pattern.
  - Change the pattern options by clicking on the drop-down menu of the respective fields.
  - The **Starting Address** of the **Golden Pattern** is automatically assigned.
  - Change the **Starting Address** of the **Golden Pattern** by clicking on the drop-down menu.
- Select the following options as required:
  - **Byte Wide Bit Mirror** – Flips each byte in Intel, Extended Tektronix, or Motorola hexadecimal data files.

    For example, 0xCD (b1100 1101) becomes 0xB3 (b1011 0011) when this is selected.
  - **Retain Bitstream Header** – By default, Radiant Deployment Tool replaces the bitstream header information (name, version number, and date of the file) with 0xFF values.

    Selecting this option retains the header information that was generated as the header.
  - **Optimize Memory Space** – When checked, it uses the bitstream file size instead of the worst-case bitstream size.
  - **SFDP Enabled** – Enables the checking of Serial Flash Discoverable Parameter (SFDP) signature during controller SPI booting. This option must be enabled for flash devices that support SFDP.
  - **Encryption** – Enables encryption of an unencrypted bitstream. After enabling, click on the **Edit Key** button and select the encryption key to be used.
  - **ECDSA Authentication** – Enables bitstream ECDSA authentication. After enabling, select **Authentication Mode** from the drop-down list, then load the **Public Key** and the **Private Key**.
  - **RSA Authentication** – Enables bitstream RSA Authentication. After enabling, select **Authentication Mode** from the drop-down list, then load the **Public Key** and the **Private Key**.
- Click **Next**.

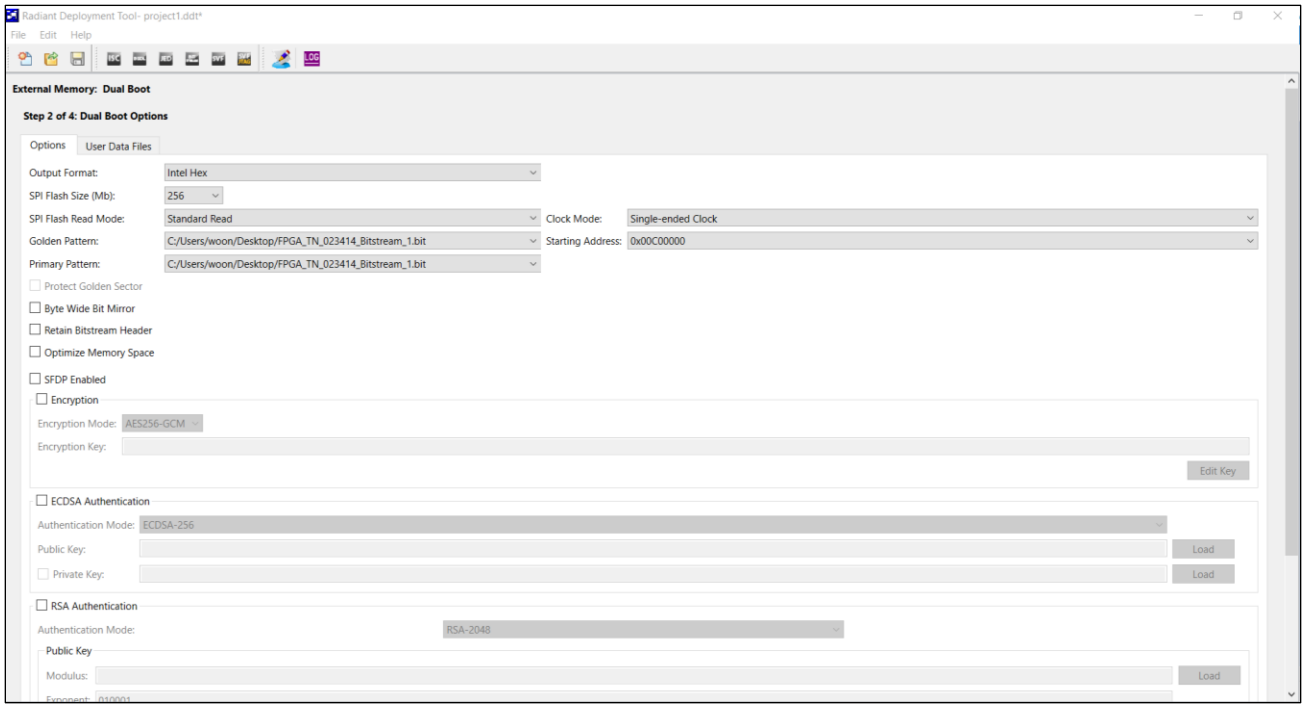


Figure 7.3. Dual Boot Options Window

Step 3 of 4: Select Output File(s) window (Figure 7.4)

- Specify the name and location of the output SPI storage device hex file in the **Output File 1** field.
- Click **Next**.

Figure 7.4. Select Output File(s) Window

Step 4 of 4: Generate Deployment window (Figure 7.5)

- Review the summary information.
- If everything is correct, click the **Generate** button.
- The **Generate Deployment** pane should indicate that the SPI storage device file was generated successfully.
- The generated .mcs file is located in the same directory as the two bitstream files.
- Save the deployment settings by selecting **File > Save**.
- To exit, select **File > Exit**.

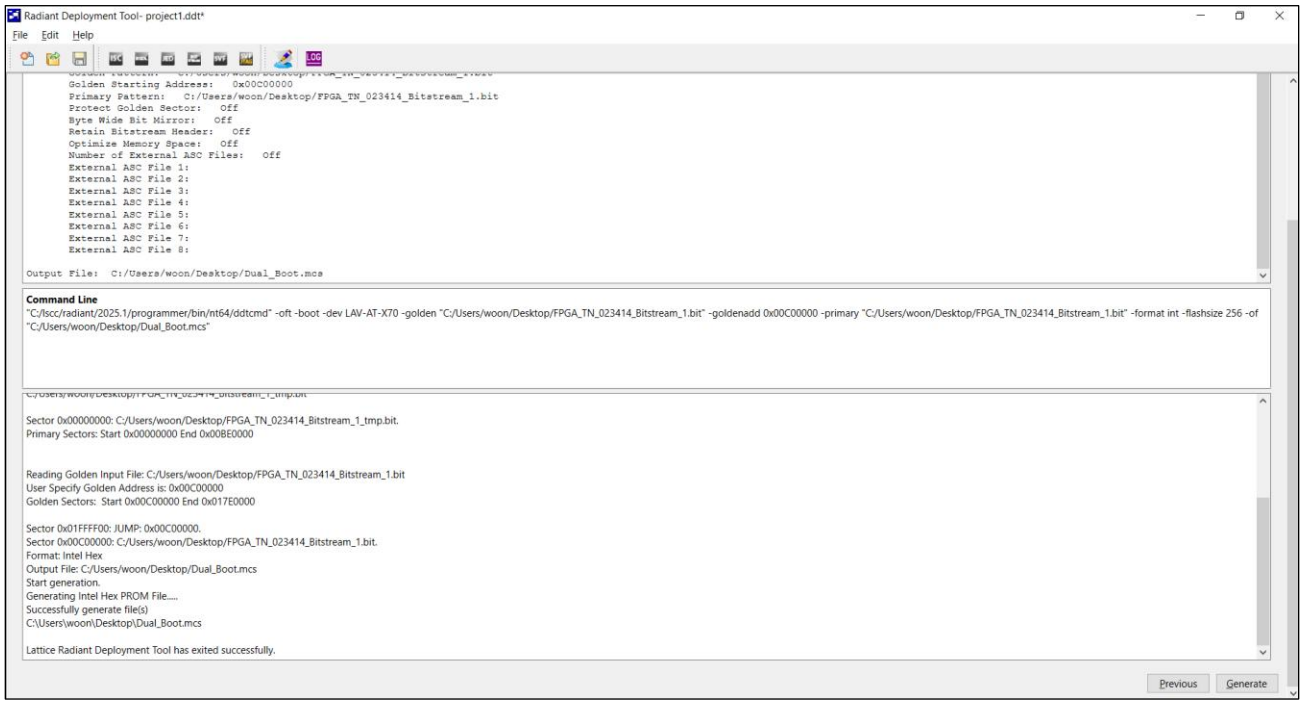


Figure 7.5. Generate Deployment Window

7.2. Using Radiant Deployment Tool to Create Ping-Pong Boot Hex File

The section provides the procedure for generating a ping-pong boot SPI storage device hex file using the Radiant Deployment Tool.

- Generate the two bitstream files in the Lattice Radiant software.
  - MCCLK_FREQ (MCLKP frequency) setting of bitstream files should not exceed the external SPI storage device normal/standard read speed.
  - MCCLK_FREQ can be configured using the Global tab of the Device Constraint Editor in the Lattice Radiant software.
- Invoke the Lattice Radiant Deployment Tool by choosing **Tools > Deployment Tool** from the Lattice Radiant Programmer.
- In the Radiant Deployment Tool window, select **External Memory** as the **Function Type** and select **Ping-Pong Boot** as the **Output File Type** (Figure 7.6).
- Click **OK**.

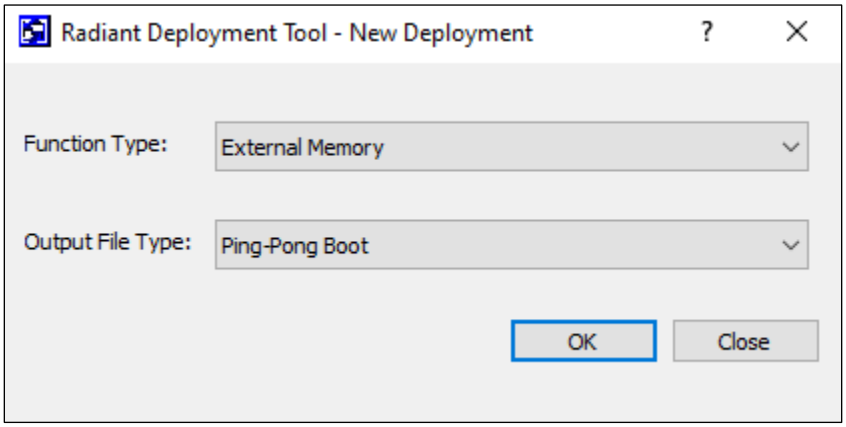


Figure 7.6. Creating New Deployment for Ping-Pong Boot File

Step 1 of 4: Select Input File(s) window (Figure 7.7)

- Click on the **…** button in the **File Name** fields to browse and select the two bitstream files to be used to create the SPI storage device hex file.
- The **Device Family** and **Device** fields automatically populate based on the bitstream files selected.
- Click **Next**.

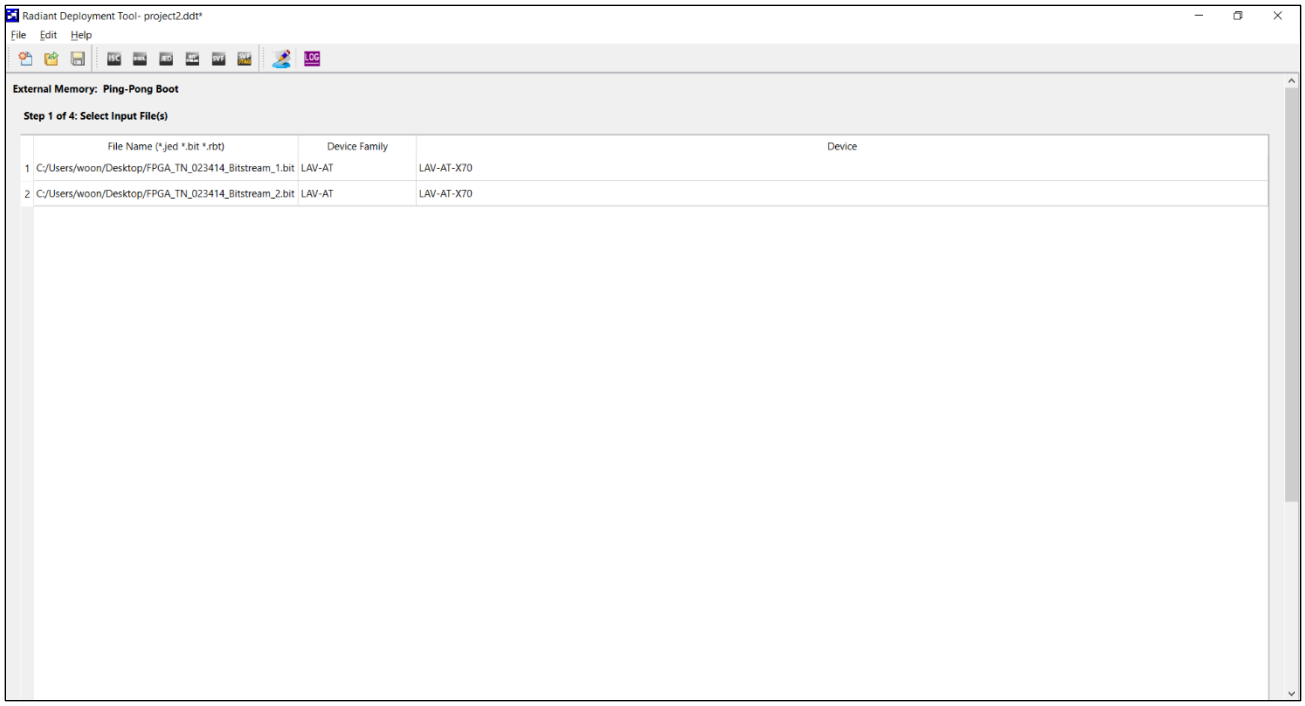


Figure 7.7. Select Input File(s) Window

Step 2 of 4: Ping-pong Boot Options window (Figure 7.8)

- Select the **Output Format** (Intel Hex, Motorola Hex, or Extended Tektronix Hex).
- Select the **SPI Flash Size (Mb)** (4, 8, 16, 32, 64, 128, 256, 512, or 1024 Mb).
- Select **SPI Flash Read Mode** (Standard Read, Fast Read, Dual I/O SPI Flash Read, or Quad I/O SPI Flash Read).
- Single-ended Clock is the only available option for **Clock Mode** field.
- The Radiant Deployment Tool automatically assigns the bitstream files selected in Step 1 to be used for the primary pattern and secondary pattern.
  - Change the pattern options by clicking on the drop-down menu of the respective fields.
  - The **Starting Address** fields of the **Primary Pattern** and **Secondary Pattern** are automatically assigned.
  - Change the **Starting Address** of the patterns by clicking on the drop-down menu.
- Select the following options as required:
  - **Generate Jump Table Only** – Generates jump table only.
  - **Byte Wide Bit Mirror** – Flips each byte in Intel, Extended Tektronix, or Motorola hexadecimal data files.

    For example, 0xCD (b1100 1101) becomes 0xB3 (b1011 0011) when this is selected.
  - **Retain Bitstream Header** – By default, Radiant Deployment Tool replaces the bitstream header information (name, version number, and date of the file) with 0xFF values.

    Selecting this option retains the header information that is generated as the header.
  - **SFDP Enabled** – Enables the checking of SFDP signature during controller SPI booting. This option must be enabled for flash devices that support SFDP.
  - **Encryption** – Enables encryption of an unencrypted bitstream. After enabling, click on the **Edit Key** button and select the encryption key to be used.
  - **ECDSA Authentication** – Enables bitstream ECDSA authentication. After enabling, select **Authentication Mode** from the drop-down list, then load the **Public Key** and the **Private Key**.
  - **RSA Authentication** – Enables bitstream RSA Authentication. After enabling, select **Authentication Mode** from the drop-down list, then load the **Public Key** and the **Private Key**.
- Click **Next**.

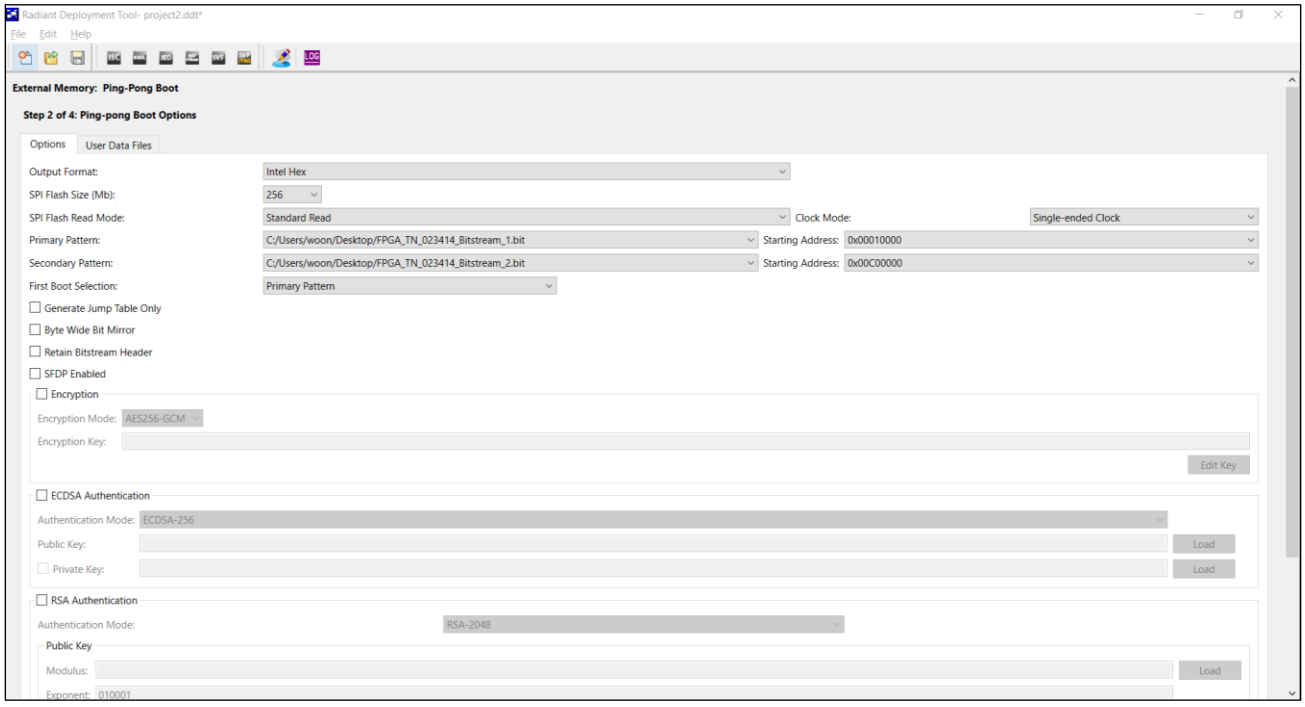


Figure 7.8. Ping-Pong Boot Options Window

Step 3 of 4: Select Output File(s) window (Figure 7.9)

- Specify the name and location of the output SPI storage device hex file in the **Output File 1** field.
- Click **Next**.

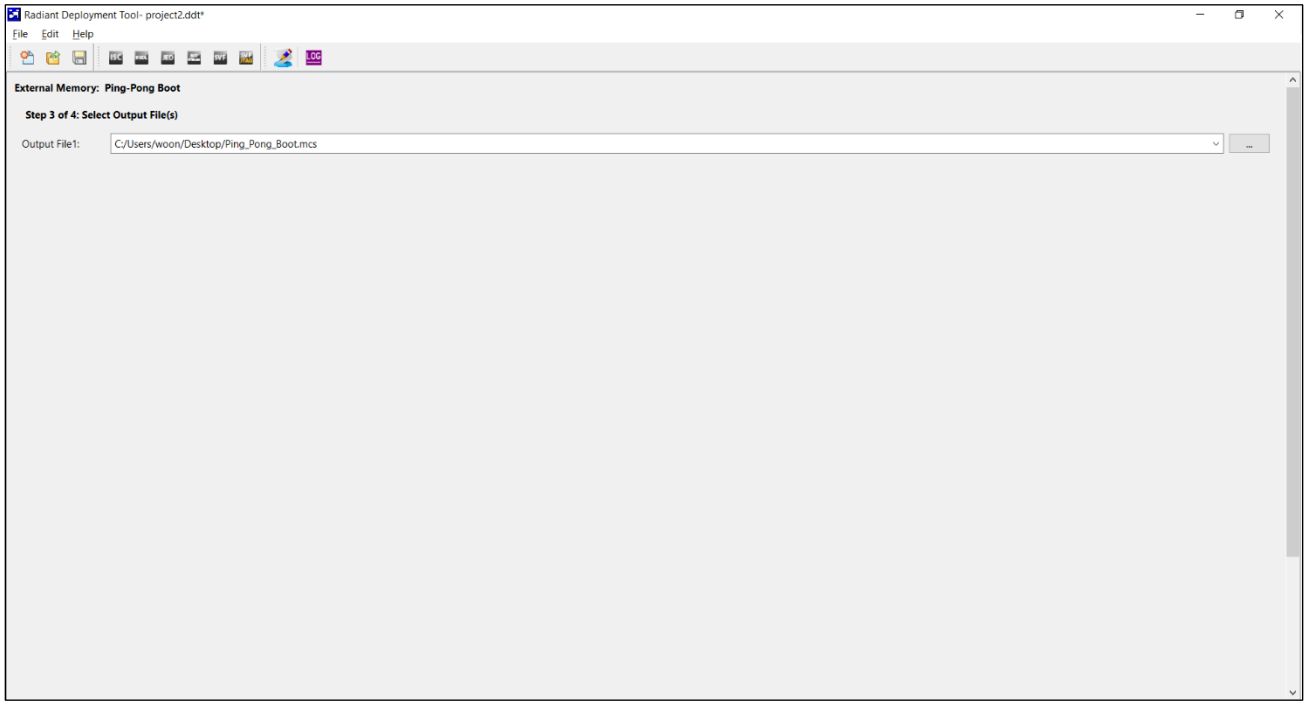


Figure 7.9. Select Output File(s) Window

Step 4 of 4: Generate Deployment window (Figure 7.10).

- Review the summary information.
- If everything is correct, click the **Generate** button.
- The **Generate Deployment** pane should indicate that the SPI storage device file is generated successfully.
- The generated .mcs file is located in the same directory as the two bitstream files
- Save the deployment settings by selecting **File > Save** or **File > Save As…**.
- To exit, select **File > Exit**.

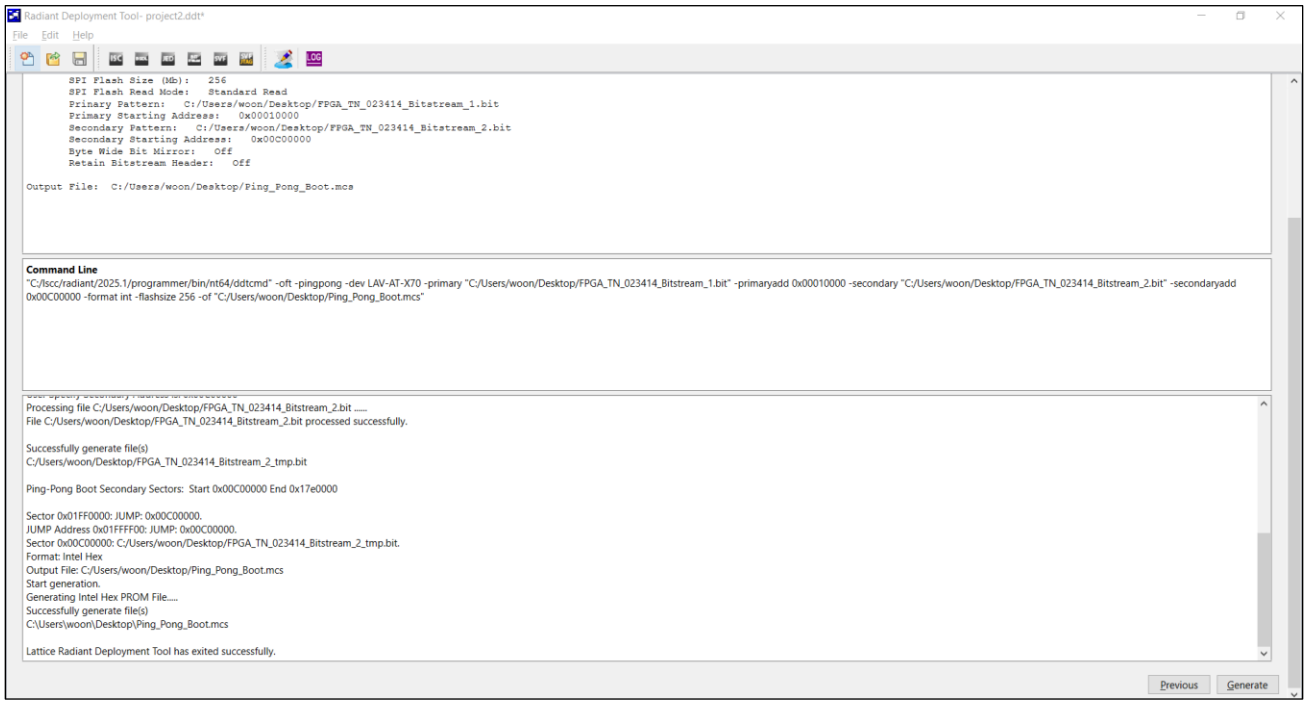


Figure 7.10. Generate Deployment Window

7.3. Using Radiant Deployment Tool to Create Multi-Boot Hex File

The section provides the procedure for generating a multi-boot SPI storage device hex file using the Radiant Deployment Tool.

- Generate the number of bitstream files needed in the Lattice Radiant software.
  - MCCLK_FREQ (MCLKP frequency) setting of bitstream files should not exceed the external SPI storage device normal/standard read speed.
  - MULTI_BOOT_MODE must be set to ENABLE to enable multi-boot mode.
  - MULTI_BOOT_SEL option selects the booting address for the multiple boot event when MULTI_BOOT_MODE is set to ENABLE.
    - STATIC (default): Use the value stored in the MultiBoot Offset register as the booting address. The MultiBoot Offset register value is set according to the bitstream pattern boot sequence in the Lattice Radiant Deployment Tool. Refer to the MultiBoot Offset section in the Lattice Avant sysCONFIG User Guide (FPGA-TN-02299) for more information about the MultiBoot Offset register.
    - DYNAMIC: Use the multi-boot address register programmed through the CONFIG_LMMI interface.
  - MCCLK_FREQ, MULTI_BOOT_MODE, and MULTI_BOOT_SEL can be configured using the Global tab of the Device Constraint Editor in the Lattice Radiant software (Figure 7.11).

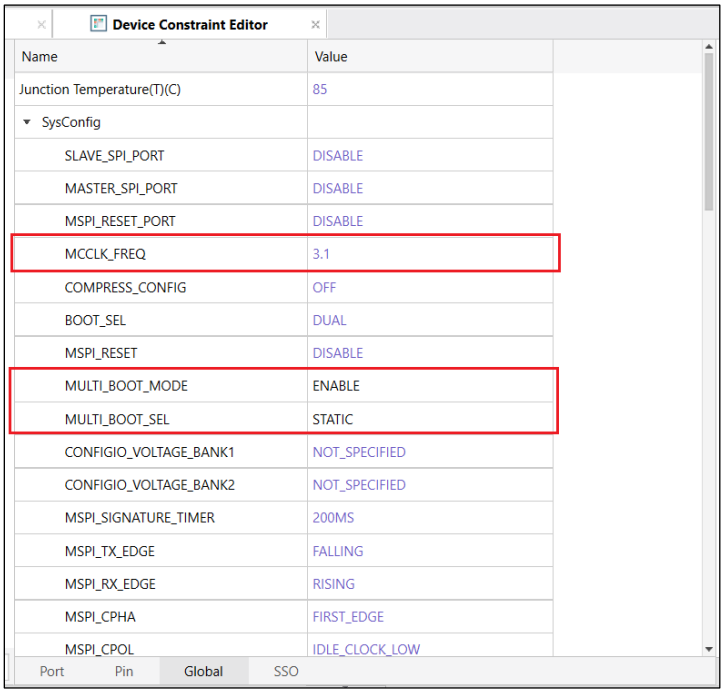


Figure 7.11. Configuring MCCLK_FREQ, MULTI_BOOT_MODE, and MULTI_BOOT_SEL in Device Constraint Editor

- Invoke the Lattice Radiant Deployment Tool by choosing **Tools > Deployment Tool** from the Lattice Radiant Programmer.
- In the Radiant Deployment Tool window, select **External Memory** as the **Function Type** and select **Advanced SPI Flash** as the **Output File Type** (Figure 7.12).
- Click **OK**.

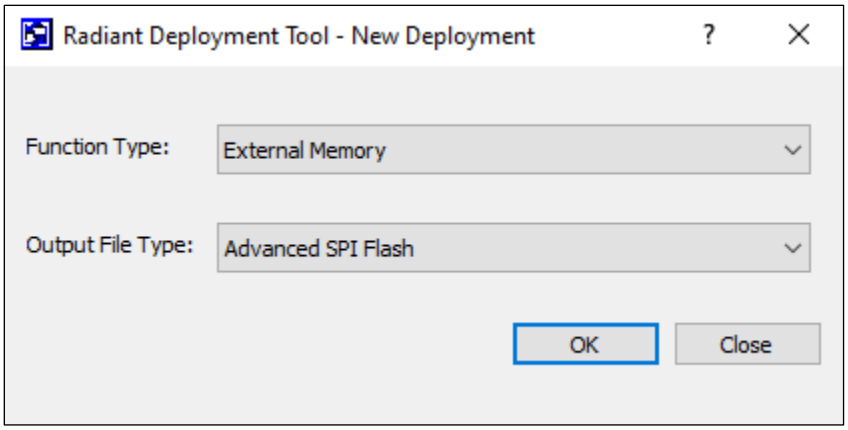


Figure 7.12. Creating New Deployment for Multi-Boot File

Step 1 of 4: Select Input File(s) window (Figure 7.13)

- Click on the **…** button in the **File Name** fields to browse and select the primary bitstream file to be used to create the SPI storage device hex file.
- The **Device Family** and **Device** fields automatically populate based on the bitstream file selected.
- Click **Next**.

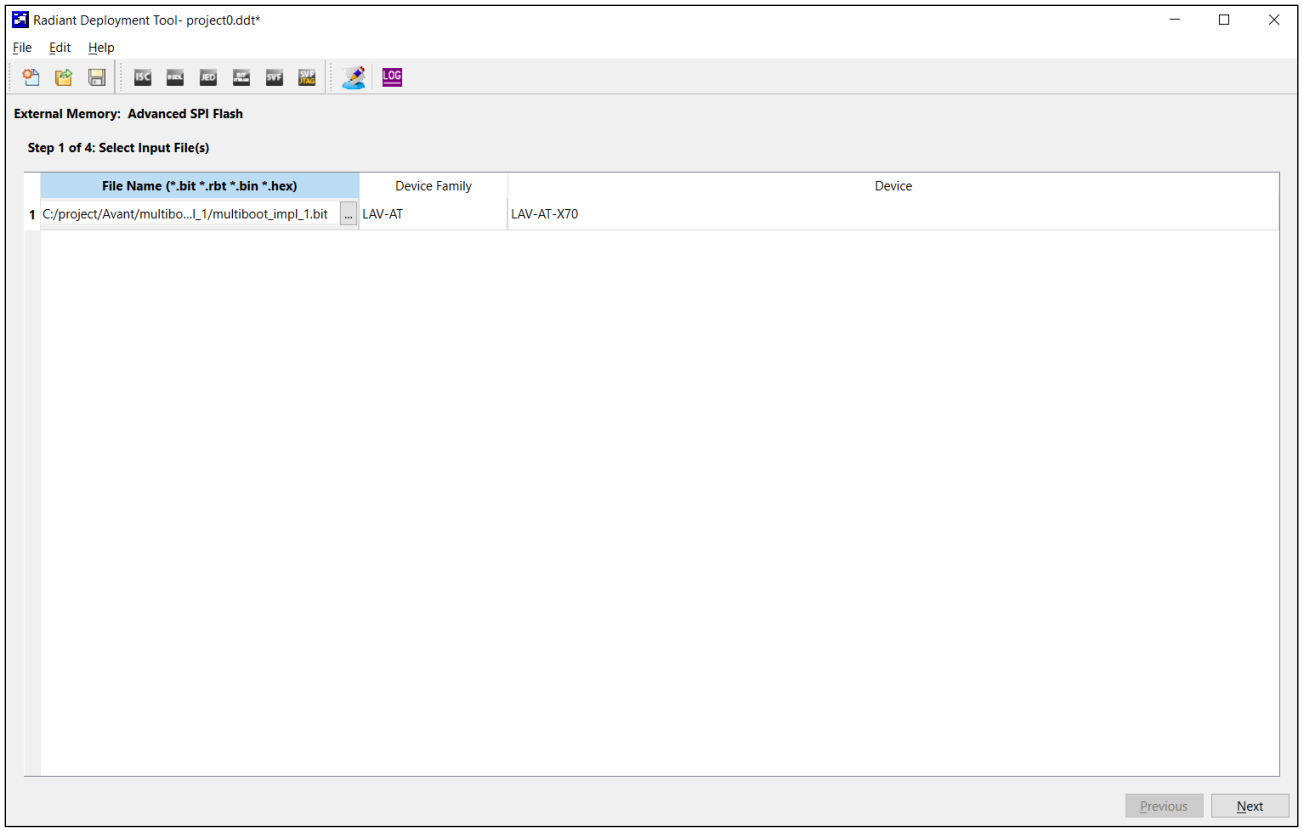


Figure 7.13. Select Input File(s) Window

Step 2 of 4: Advanced SPI Flash Options window

- Edit the following options in the **Options** tab (Figure 7.14):
  - Select the **Output Format** (Intel Hex, Motorola Hex, or Extended Tektronix Hex).
  - Select the **SPI Flash Size (Mb)** (4, 8, 16, 32, 64, 128, 256, 512, or 1024 Mb).
  - Select **SPI Flash Read Mode** (Standard Read, Fast Read, Dual I/O SPI Flash Read, or Quad I/O SPI Flash Read).
  - Single-ended Clock is the only available option for **Clock Mode** field.
  - Select the following options as required:
    - **Byte Wide Bit Mirror** – Flips each byte in Intel, Extended Tektronix, or Motorola hexadecimal data files. For example, 0xCD (b1100 1101) becomes 0xB3 (b1011 0011) when this is selected.
    - **Retain Bitstream Header** – By default, Radiant Deployment Tool replaces the bitstream header information (name, version number, and date of the file) with 0xFF values. Selecting this option retains the header information that is generated as the header.
    - **Optimize Memory Space** – When checked, it uses the bitstream file size instead of the worst-case bitstream size.
    - **SFDP Enabled** – Enables the checking of SFDP signature during controller SPI booting. This option must be enabled for flash devices that support SFDP.
    - **Encryption** – Enables encryption of an unencrypted bitstream. After enabling, click on the **Edit Key** button and select the encryption key to be used.
    - **ECDSA Authentication** – Enables bitstream ECDSA authentication. After enabling, select **Authentication Mode** from the drop-down list, then load the **Public Key** and the **Private Key**.
    - **RSA Authentication** – Enables bitstream RSA Authentication. After enabling, select **Authentication Mode** from the drop-down list, then load the **Public Key** and the **Private Key**.

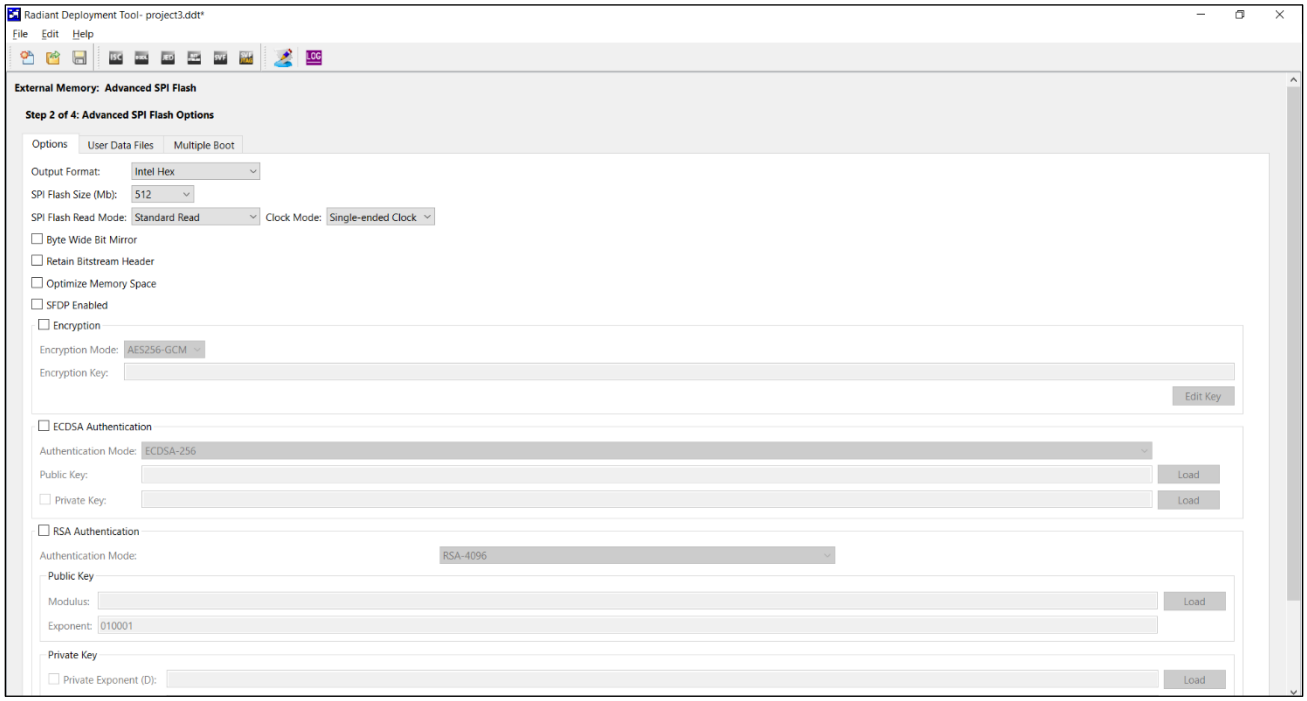


Figure 7.14. Advanced SPI Flash Options Window – Options Tab

- Edit the following options in the **Multiple Boot** tab (Figure 7.15):
  - Click on the **Multiple Boot** check box to enable multi-boot mode.
  - Click on the **Golden Pattern …** button and select the bitstream file.
    - The starting address of the golden pattern is automatically populated.
    - Change the **Starting Address** of the patterns by clicking on the drop-down menu.
  - **Protect Golden Sector** – By default, the golden sector, where the golden pattern is stored, is located immediately after the primary sector to save SPI flash space. When this option is selected, the golden pattern location is moved to the first sector in the upper half of the SPI flash. The new location is reflected in the **Golden Patten Starting Address** field. This protects the golden pattern from accidental erase/reprogram by protecting the upper half of the SPI flash when it is programmed.
  - In the **Multi_Boot_Sel Options** field, select either **Static** or **Dynamic** from the drop-down list.
  - In the **Number of Alternate Patterns** field, select the number of alternate patterns to include through the drop-down menu, selecting between one to four alternate patterns.
  - In the **Alternate Pattern 1** field, click on the **...** button to select the first alternate pattern.
    - The **Starting Address** of **Alternate Pattern 1** is automatically populated.
    - You can change the **Starting Address** of **Alternate Pattern 1** by clicking on the drop-down menu.
    - The **Next Alternate Pattern to Configure** field is automatically populated. This is the pattern that is loaded during the next PROGRAMN/REFRESH event. You can change the pattern by clicking on the drop-down menu.
  - Configure the remaining alternate patterns as necessary.
- Click **Next**.

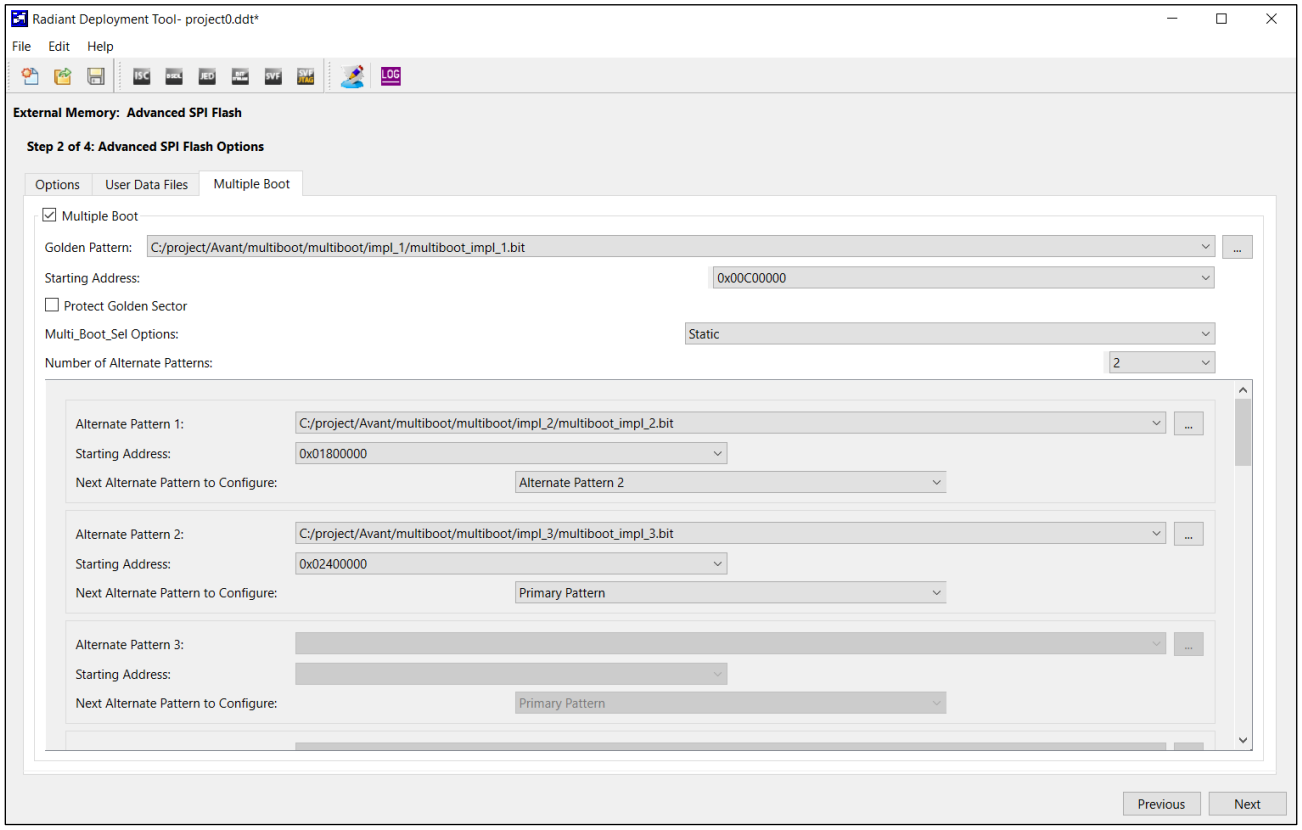


Figure 7.15. Advanced SPI Flash Options Window – Multiple Boot Tab

Step 3 of 4: Select Output File(s) window (Figure 7.16)

- Specify the name and location of the output SPI storage device hex file in the **Output File 1** field.
- Click **Next**.

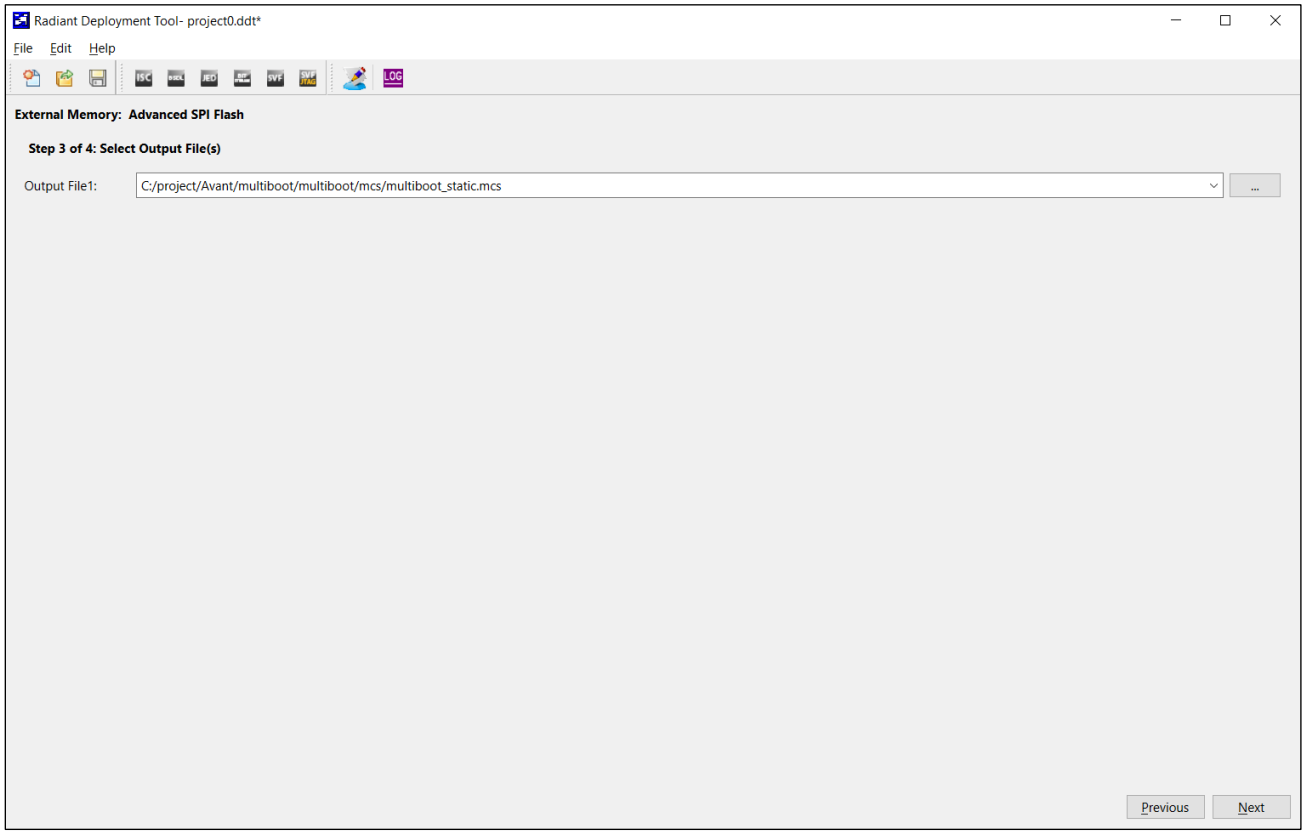


Figure 7.16. Select Output File(s) Window

Step 4 of 4: Generate Deployment window (Figure 7.17)

- Review the summary information.
- If everything is correct, click the **Generate** button.
- The **Generate Deployment** pane should indicate that the SPI storage device file is generated successfully.
- The generated .mcs file is located in the directory as specified in step 3 of 4.
- Save the deployment settings by selecting **File > Save** or **File > Save As…**.
- To exit, select **File > Exit**.

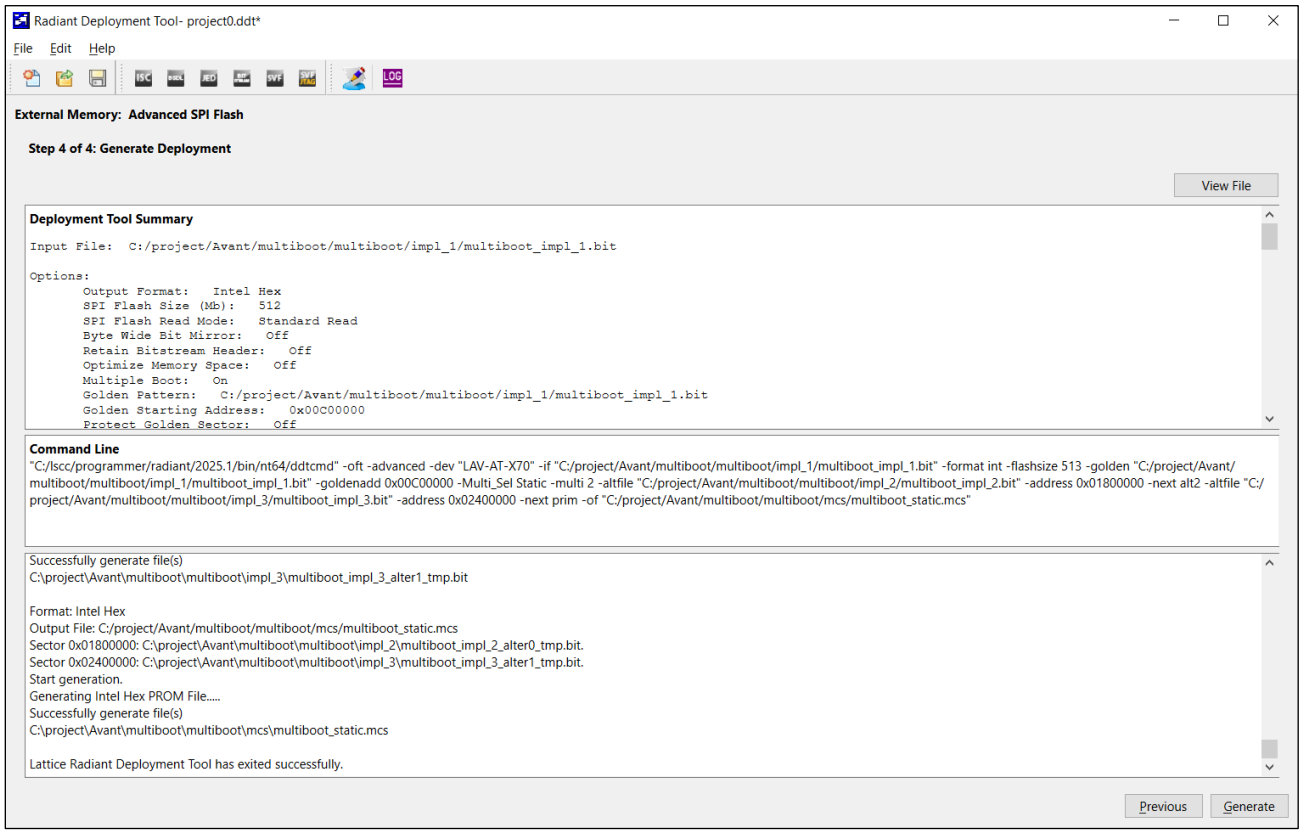


Figure 7.17. Generate Deployment Window

8. Programming Dual-Boot, Ping-Pong Boot, or Multi-Boot Pattern into an External SPI Storage Device

The following procedure is for programming a dual-boot, ping-pong boot, or multi-boot pattern into the SPI storage device using the Radiant Programmer:

1. Connect power to the board and connect a download cable from the board to the PC.
2. Invoke Radiant Programmer 2023.2 or later using one of the following methods:
   - In Radiant Software window, select **Tools > Programmer**.
   - In Radiant Software window, select the **Programmer** icon ( ) in the Radiant toolbar.
   - In the Windows Start menu, select **Start > Lattice Radiant Software 2023.2 > Radiant Programmer**.
3. The Radiant Programmer – Getting Started window opens (Figure 8.1).
   - Select **Create a new project from a scan**, **Create a new blank project**, or **Open an existing programmer project**.
   - If **Create a new project from a scan** is selected, click **Detect Cable** to scan the PC to determine what cable is connected or manually select the type of **Cable** and **Port**.
   - Click **OK**.

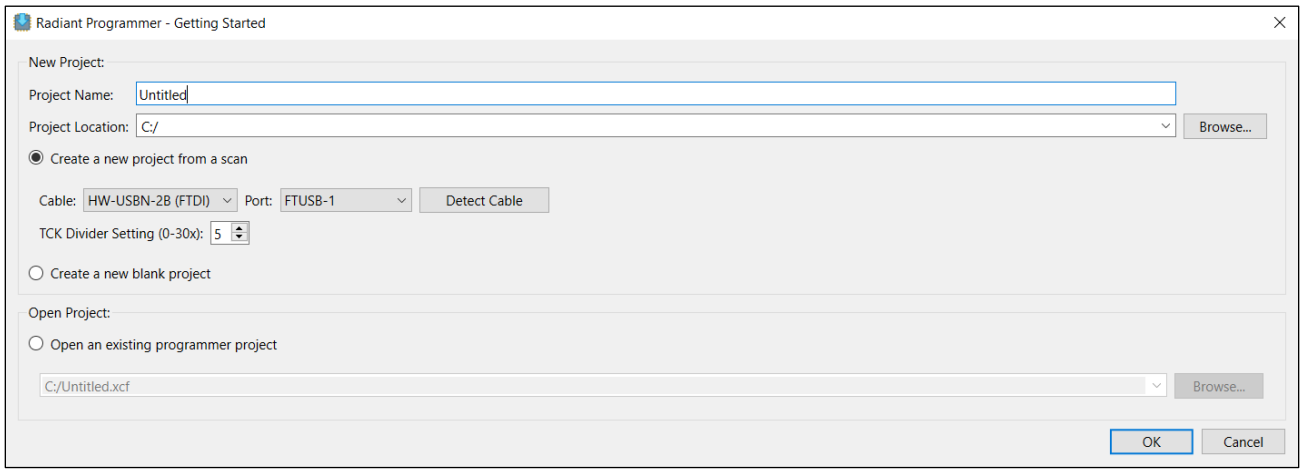


Figure 8.1. Radiant Programmer – Getting Started Window

4. The Radiant Programmer main window opens listing the device as found in a scan if applicable (Figure 8.2).

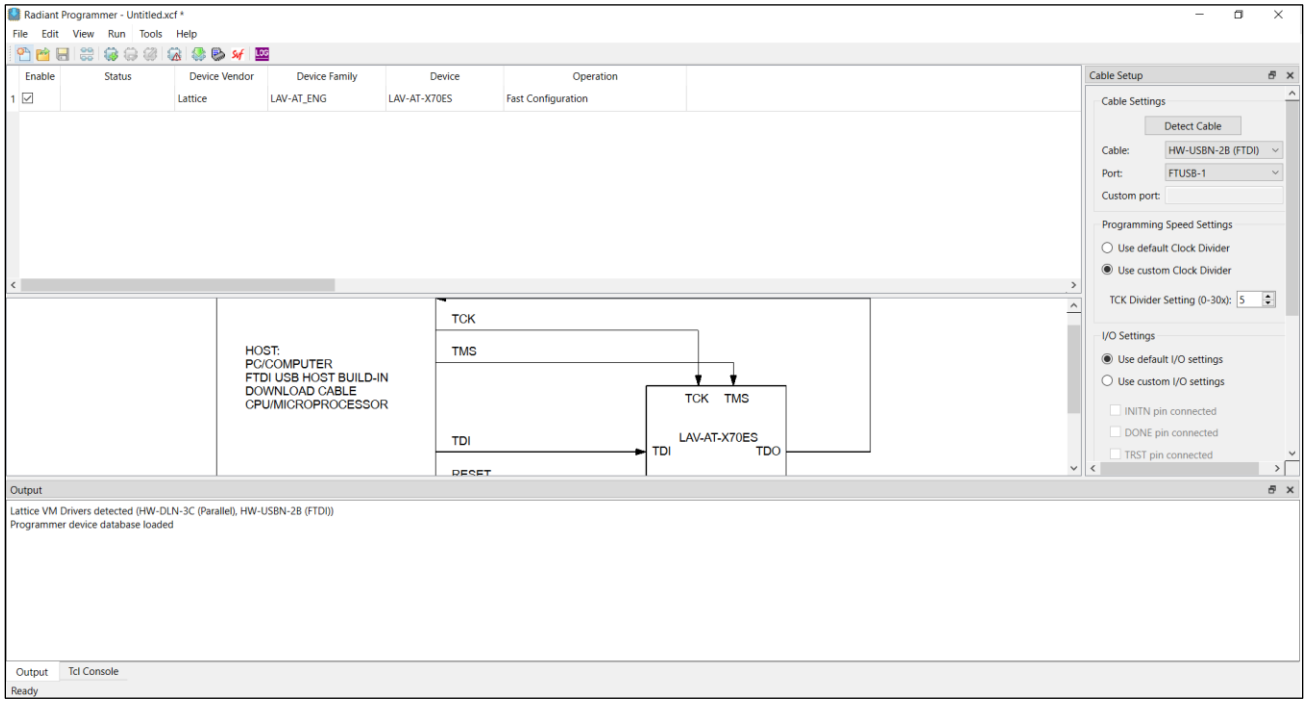


Figure 8.2. Radiant Programmer – Main Window

5. Double-click on the **Operation** field to open the **Device Properties** window.
6. Set the **Device Properties** fields to the following (Figure 8.3):
   - For **Target Memory**, select **External SPI Flash Memory (SPI Flash)**.
   - For **Port Interface**, select **JTAG2SPI**.
   - For **Access Mode**, select **Direct Programming**.
   - For **Operation**, select **Erase, Program, Verify**.
   - For **Programming file**, browse to select the .mcs file to be programmed.
   - In the **SPI Flash Options** field, specify the **Family**, **Vendor**, **Device**, and **Package** of the flash device used on the board.
     - **Family** – SPI Serial Flash
     - **Vendor** – Micron
     - **Device** – MT35XU512ABA1G12_0SIT
     - **Package** – 24-ball T-PBGA
   - For **Data file size (Bytes)**, click on the **Load from File** button.
   - Click **OK**.

   Note: This example shows the configuration for programming of the Avant-X Versa board.

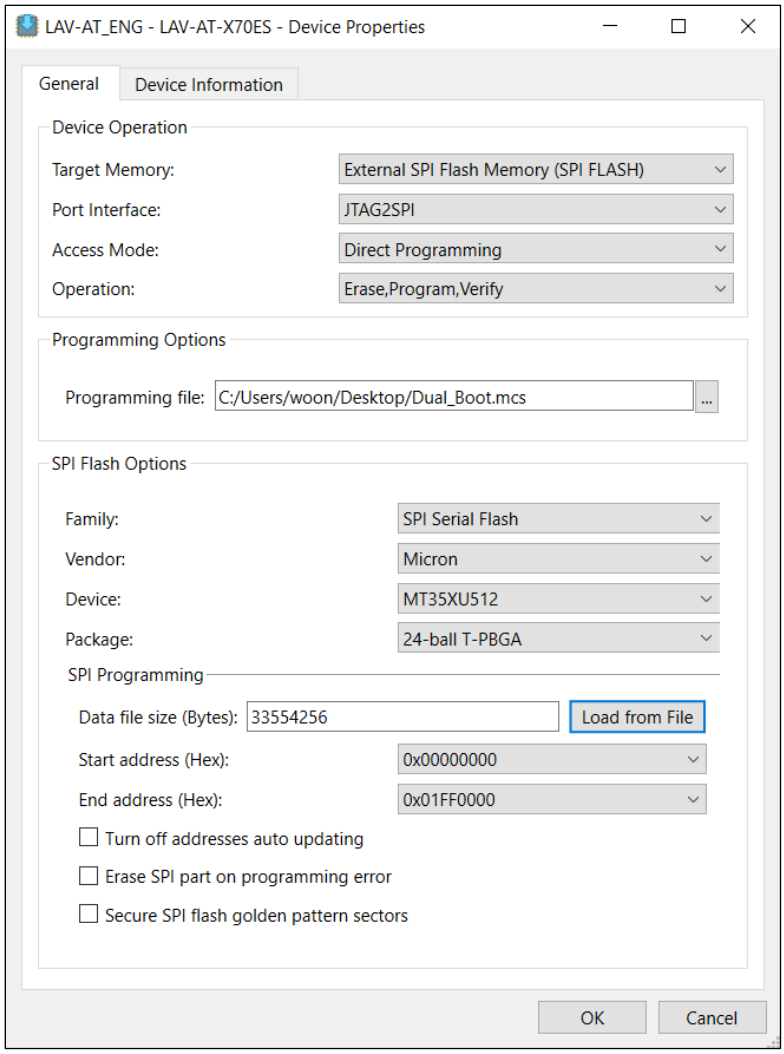


Figure 8.3. Radiant Programmer – Device Properties

7. Program the external SPI storage device using one of the following methods:
   - In the main Radiant Programmer window, select **Run > Program Device**.
   - In the main Radiant Programmer window, select the **Program Device** icon ( ) in the toolbar.

References

- Avant-E web page
- Avant-G web page
- Avant-X web page

A variety of technical notes for the Lattice Avant platform are available.

- High-Speed PCB Design Considerations (FPGA-TN-02178)
- Lattice Avant Embedded Memory User Guide (FPGA-TN-02289)
- Lattice Avant Hardware Checklist (FPGA-TN-02317)
- Lattice Avant High-Speed I/O and External Memory Interface (FPGA-TN-02300)
- Lattice Avant Power User Guide (FPGA-TN-02291)
- Lattice Avant sysCLOCK PLL Design and User Guide (FPGA-TN-02298)
- Lattice Avant sysCONFIG User Guide (FPGA-TN-02299)
- Lattice Avant sysDSP User Guide (FPGA-TN-02293)
- Lattice Avant sysI/O User Guide (FPGA-TN-02297)
- Lattice Memory Mapped Interface and Lattice Interrupt Interface User Guide (FPGA-UG-02039)
- sub-LVDS Signaling Using Lattice Devices (FPGA-TN-02028)
- Thermal Management (FPGA-TN-02044)
- Using TraceID (FPGA-TN-02084)

For more information on Lattice Avant-related IP, reference designs, and board documents, refer to the following pages:

- SGMII and Gb Ethernet PCS IP Core – Lattice Radiant Software (FPGA-IPUG-02077)
- Lattice Avant OSC Module User Guide (FPGA-IPUG-02184)
- Lattice Avant Device Multi-Boot Reference Design (FPGA-RD-02321)
- Avant-E Evaluation Board – User Guide (FPGA-EB-02057)
- Avant-G/X Versa Board – User Guide (FPGA-EB-02063)
- IP and Reference Designs for Avant
- Development Kits and Boards for Avant

For further information on interface standards refer to the following websites:

- JEDEC Standards (LVTTL, LVCMOS, SSTL) – www.jedec.org
- PCI – www.pcisig.com

Other references:

- Lattice Insights for Lattice Semiconductor training courses and learning plans
- Lattice Radiant FPGA design software

Technical Support Assistance

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

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

Revision History

Revision 0.86, September 2025

Section	Change Summary
Abbreviations in This Document	Added *CFG*, *FPGA*, *FSM*, *ID*, *JTAG*, *LMMI*, *OSC*, *PC*, and *RAM*.
Definition of Terms	Updated definition of *Alternative Boot Pattern* to include reference to *alternate pattern*.
Dual-Boot Mode	Added note regarding 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row in the Dual-Boot Flow section.
Ping-Pong Boot Mode	Added note regarding 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row in the Ping-Pong Boot Flow section.

Multi-Boot Mode	• In the Multi-Boot Flow section:
	  • Added Figure 6.2. Static Multi-Boot Mode Flow and Figure 6.3. Dynamic Multi-Boot Mode Flow.
	  • Made minor editorial changes to description.
	  • Added statement on condition when golden pattern configuration fails.
	  • Added note regarding 32-bit MSPI Address and 32-bit MSPI Commands items in the feature row.
	  • Updated description of STATIC setting for the MULTI_BOOT_SEL option.
	  • Added description for the Multiboot Offset option.
	• Added the Implementation of Static or Dynamic Multi-Boot Mode section.

Creating an SPI Storage Device Hex File	• In the Using Radiant Deployment Tool to Create Multi-Boot Hex File section:
	  • Updated Figure 7.13. Select Input File(s) Window, Figure 7.15. Advanced SPI Flash Options Window – Multiple Boot Tab, Figure 7.16. Select Output File(s) Window, and Figure 7.17. Generate Deployment Window.
	  • Added Multi_Boot_Sel Options under Multiple Boot tab edits in step 2 of 4.

References	• Added OSC Module IP user guide and Multi-Boot reference design. • Updated IP and Reference Designs for Avant and Development Kits and Boards for Avant links.

Revision 0.85, July 2025

Section	Change Summary

Resources	• In Table 3.1. Bitstream Size versus Recommended SPI Storage Size (Single Bitstream Boot Mode):
	  • Updated bitstream sizes for LAV-AT-E/G/X30 (maximum EBR only), LAV-AT-E/G/X50, and LAV-AT-E/G/X70.
	  • Updated recommended SPI storage device sizes for LAV-AT-E/G/X30 (maximum EBR only) and LAV-AT-E/G/X50 (maximum EBR only).
	• In Table 3.2. Bitstream Size versus Recommended SPI Storage Size (Dual-Boot Mode, Ping-Pong Boot Mode):
	  • Updated bitstream sizes for LAV-AT-E/G/X30 (maximum EBR only), LAV-AT-E/G/X50, and LAV-AT-E/G/X70.
	  • Updated recommended SPI storage device sizes for LAV-AT-E/G/X30 (maximum EBR only) and LAV-AT-E/G/X50 (maximum EBR only).
	• In Table 3.3. Bitstream Size versus Recommended SPI Storage Size (Multi-Boot Mode):
	  • Updated bitstream sizes for LAV-AT-E/G/X30 (maximum EBR only), LAV-AT-E/G/X50, and LAV-AT-E/G/X70.
	  • Updated recommended SPI storage device size for LAV-AT-E/G/X30 with the following:
	  • Maximum EBR; 4 bitstream patterns
	  • Updated recommended SPI storage device sizes for LAV-AT-E/G/X50 with the following:
	  • No EBR; 3 and 6 bitstream patterns
	  • Maximum EBR scenario; 4 and 5 bitstream patterns
	  • Updated recommended SPI storage device size for LAV-AT-E/G/X70 with the following:
	  • No EBR; 5 bitstream patterns

Revision 0.84, December 2024

Section	Change Summary
All	• Made minor editorial changes. • Updated term from SPI flash device to SPI storage device throughout document. • Updated term from SPI master clock frequency to MCLKP frequency.
Abbreviations in This Document	• Updated section title, description, and table header. • Moved definition of terms into the Definition of Terms section and retained abbreviations in this section. • Added *ECDSA*, *MSPI*, *RSA*, *SFDP*, *SRAM*, *T-PBGA*, and *xSPI*.
Introduction	Updated description.

Definition of Terms	• Added new section.
	• For terms moved from the Abbreviations in This Document section:
	  • Updated definition of *Alternative Boot Pattern*, *Binary Hex Data File*, *Bitstream Data File*, *Dual-Boot*, *Flash Lock*, *Hex Data File*, *Multi-Boot*, *Refresh*, and *Sector (Block)*.
	  • Removed *Configure*, *FD-SOI*, and *Program*.
	  • Added *Configuration*, *MCLKP*, *Ping-Pong Boot*, and *Programming*.
	  • Updated term from *Golden Boot* to *Golden Pattern* and definition.
	  • Updated term from *Primary Boot* to *Primary Pattern* and definition.

Resources	Updated description and revamped tables.

Dual-Boot Mode	• Updated section title and description.
	• Updated Figure 4.1. Dual-Boot Flow Diagram.
	• In the Dual-Boot Flow section:
	  • Updated section title and descriptions.
	  • Removed the Bitstream Preamble Codes table.
	  • Added default setting designation for timer value in Table 4.1. Control Register 1 [2:0] – MSPI Preamble Timer Count.

Ping-Pong Boot Mode	• Updated section title and description. • Updated Figure 5.1. Ping-Pong Boot Flow Diagram. • In the Ping-Pong Boot Flow section: • Updated section title and descriptions.
Multi-Boot Mode	• Updated section title and description. • Added Figure 6.1. Multi-Boot Flow Diagram. • Added the Multi-Boot Flow section.

Creating an SPI Storage Device Hex File	• Updated term from PROM to SPI storage device.
	• Updated section title and description.
	• In the Using Radiant Deployment Tool to Create a Dual-Boot Hex File section:
	  • Updated Figure 7.2. Select Input File(s) Window through Figure 7.5. Generate Deployment Window.
	  • Added SFDP Enabled, Encryption, ECDSA Authentication, and RSA Authentication options and descriptions for the Dual Boot Options window.
	• In the Using Radiant Deployment Tool to Create Ping-Pong Boot Hex File section:
	  • Updated Figure 7.7. Select Input File(s) Window through Figure 7.10. Generate Deployment Window.
	  • Added Encryption, ECDSA Authentication, and RSA Authentication options and descriptions for the Ping-Pong Boot Options window.
	• In the Using Radiant Deployment Tool to Create Multi-Boot Hex File section:
	  • Updated and added conditions to generate the bitstream files.
	  • Rearranged some figures.
	  • Added Figure 7.11. Configuring MCCLK_FREQ, MULTI_BOOT_MODE, and MULTI_BOOT_SEL in Device Constraint Editor.
	  • Updated Figure 7.13. Select Input File(s) Window through Figure 7.17. Generate Deployment Window.
	  • Added ECDSA Authentication and RSA Authentication options and descriptions for the

Section	Change Summary
	Advanced SPI Flash Options window. • Updated step 4 of 4 description regarding .mcs file.
Programming Dual-Boot, Ping-Pong Boot, or Multi-Boot Pattern into an External SPI Storage Device	• Updated section title and descriptions. • Rearranged and updated Figure 8.1. Radiant Programmer – Getting Started Window through Figure 8.3. Radiant Programmer – Device Properties. • Updated flash device used from WinBond to Micron. • Updated board to Avant-X Versa board.
References	Added Avant-G/X Versa Board – User Guide.

Revision 0.83, July 2024

Section	Change Summary
Resources	Newly added Table 2.3. Maximum Configuration Bitstream Size – Multi-Boot Mode.
Avant Ping-Pong Boot Mode	Newly added this section.
Avant Multi-Boot Mode	Newl added this section.

Creating a PROM File	• In the Using Radiant Deployment Tool to Create a Dual Boot Hex File section:
	  • added *Click on the … button* in the File Name fields to browse and select bitstream files for creating PROM hex file in Step 1 of 4;
	  • added *Single-ended Clock is the only available option for Clock Mode field* to Step 2 of 4;
	  • added *The generated .mcs file is located in the same directory as the two bitstream files* to Step 4 of 4.
	• Added the Using Radiant Deployment Tool to Create Ping-Pong Boot Hex File section;
	• Added the Using Radiant Deployment Tool to Create Multi-Boot Hex File section.

Programming Dual Boot, Ping-Pong Boot, or Multi-Boot Pattern into external SPI Flash Device	• Updated the section heading from *Programming the Dual Boot Pattern into the external SPI Flash Device* to the current. • Removed the original step 4: *The Radiant Programmer main window prompts. In the Operation field, move the cursor over the highlighted operation such as Bypass and double-click the left mouse button.* • Added step 4: *If the scan is successful, Radiant Programmer main window opens listing the device it found from the scan (Figure 7.2).* • Added *step 5: Double-click in the Operation field to open the Device Properties window*. • Updated the original step 5 from *The Device Properties window opens (Figure 5.2)* to step 6, *This example shows the configuration for programming of the Avant Evaluation REVD Board. Set the Device Properties fields to the following (Figure 7.3)*. • Added Figure 7.2. Radiant Programmer – Main Window.

Revision 0.82, November 2023

Section	Change Summary
Disclaimers	Updated this section.
Resources	Updated Table 2.1. Maximum Configuration Bitstream Size – Single Bitstream Boot Mode and Table 2.2. Maximum Configuration Bitstream Size – Dual Boot Mode reflecting the Avant Product name change to the current.
Creating a PROM File	Updated Figure 4.2. Select Input Files Window and Figure 4.5. Generate Deployment Window reflecting the Avant product name change in the software GUI.
Programming the Dual Boot Pattern into the SPI Flash Device	• Updated the software version to *2023.2*. • Updated Figure 5.2. Radiant Programmer – Device Properties reflecting the Avant product name change in the software GUI.
References	Newly added section.

Revision 0.81.1, May 2023

Section	Change Summary
Resources	Updated Table 2.1. Maximum Configuration Bitstream Size – Single Bitstream Boot Mode and Table 2.2. Maximum Configuration Bitstream Size – Dual Boot Mode adding the data for Lattice Avant G/X devices.
Avant Dual Boot Mode	Updated the whole section including the newly added Table 3.1. Bitstream Preamble Codes and Table 3.2. Control Register 1 [2:0] – Master Preamble Timer Count Value reflecting the recent Avant device behavior.
Creating a PROM File	Updated the general description removing the mentioning of external SPI Flash.
Programming the Dual Boot Pattern into the SPI Flash Device	Updated procedure upon the most recent Lattice Radiant Programmer software.

Revision 0.81, April 2023

Section	Change Summary
Inclusive Language	Newly added section.
Resources	Updated Table 2.1. Maximum Configuration Bitstream Size – Single Bitstream Boot Mode and Table 2.2. Maximum Configuration Bitstream Size – Dual Boot Mode adding the data for Lattice Avant 200E, 300E and the G/X devices.
Avant Dual Boot Mode	Updated the whole section including the newly added Table 3.1. Bitstream Preamble Codes and Table 3.2. Control Register 1 [2:0] – Master Preamble Timer Count Value reflecting the recent Avant device behavior.
Creating a PROM File	Updated the general description removing the mentioning of external SPI Flash.
Programming the Dual Boot Pattern into the SPI Flash Device	Updated procedure upon the most recent Lattice Radiant Programmer software.

Revision 0.80, November 2022

Section	Change Summary
All	Initial Preliminary release.

www.latticesemi.com