

Introduction

Platform Manager™ 2 devices are fast-reacting, programmable logic based hardware management controllers. Platform Manager 2 devices are an integrated solution combining analog sense and control elements with scalable programmable logic resources. This integrated approach allows the Platform Manager 2 to address Power Management (Power Sequencing, Voltage Monitoring, Trimming and Margining), Thermal Management (Temperature Monitoring, Fan Control, Power Control), and Control Plane functions (System Configuration, I/O Expansion, Reset Distribution, and so forth).

This technical note focuses on support for reliable in-field updates – including Dual Boot mode and background programming. The information in this technical note also applies to Platform Manager 2 designs. Platform Manager 2 designs can be built using several combinations of Lattice devices as listed in Table 1. Throughout the remainder of this document, the use of ASC refers to either the LPTM21L or L-ASC10 hardware management expander.

Table 1. Lattice Platform Manager 2 Design Options

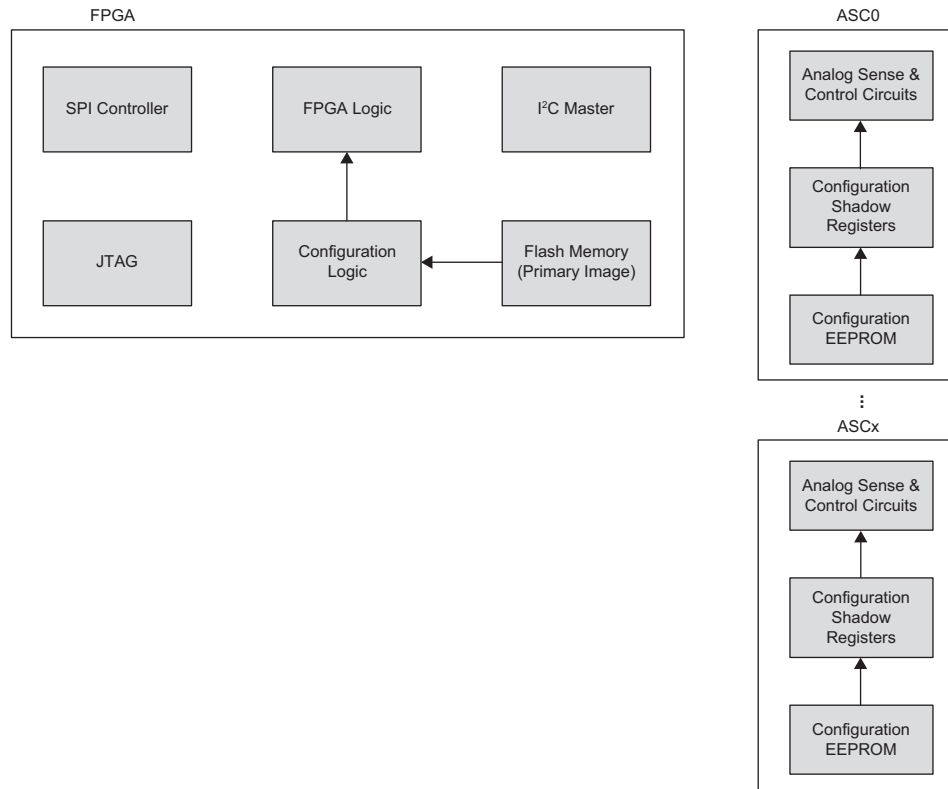
Central Hardware Manager	Hardware Management Expander	Number of Expanders Supported
LPTM21	LPTM21L or L-ASC10	0 – 3
LPTM21L	LPTM21L or L-ASC10	0 – 3
MachXO2™	LPTM21L or L-ASC10	1 – 8
MachXO3™	LPTM21L or L-ASC10	1 – 8
ECP5™	LPTM21L or L-ASC10	1 – 8

Since the ECP5 FPGA does not have internal flash memory, it supports Dual Boot and Multiple Boot using the Lattice Deployment Tool. For ECP5 based Platform Manager 2 designs, refer to TN1216, [LatticeECP3](#), [LatticeECP2/M](#), [ECP5](#) and [ECP5-5G Dual Boot and Multiple Boot Feature](#).

Theory of Operation

Devices in the Platform Manager 2 family are designed to support reliable in-field updates. The Platform Manager 2 and ASC devices normally configure at power up based on their internal non-volatile configuration memory. If the non-volatile configuration memory has been properly programmed, the FPGA will automatically configure from the internal flash memory. The ASC will configure based on its internal EEPROM memory automatically. The FPGA logic will release the ASC from safe state and the system will begin normal operation based on its primary image. The normal operation path is shown in Figure 1. See AN6091, [Powering Up and Programming Platform Manager 2 and L-ASC10](#) for more details on the I/O safe states and Power-On-Reset process.

Figure 1. Configuration from Internal Memory (After Successful Programming)

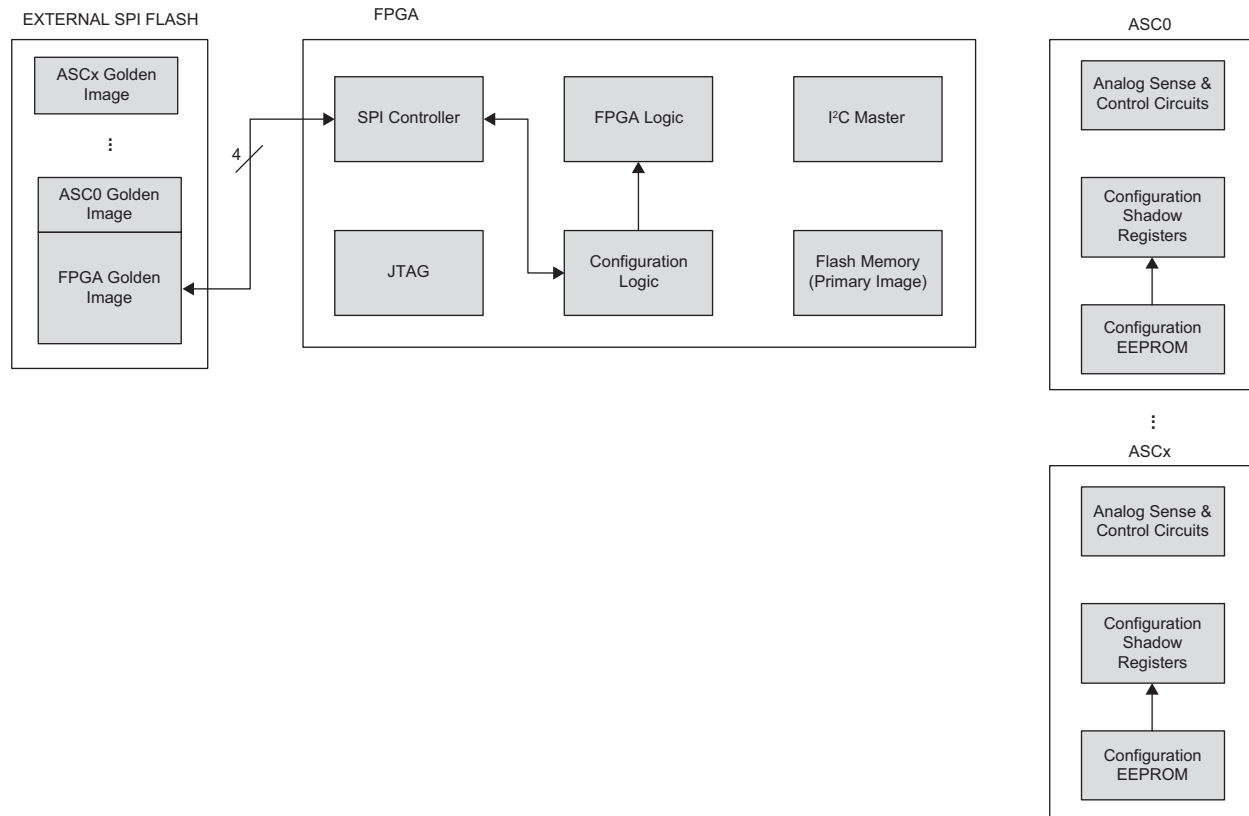


In case of a failed reprogramming or other scenario where the internal memory is not properly configured, the Dual Boot mechanism will take over (provided it has been enabled in the application). The Dual Boot operation with Platform Manager 2 is a two-stage operation. Stage 1 is the FPGA Dual Boot, and Stage 2 is the ASC Dual Boot.

Stage 1. FPGA Dual Boot

In Stage 1, the FPGA in the hardware management controller will configure from the external SPI memory. This stage uses the configuration circuitry inside the FPGA. Upon completion of Power-On-Reset (POR), the configuration circuitry will check the internal flash configuration memory. If the configuration memory has been properly programmed, the FPGA will configure based on the internal flash memory (path shown in Figure 1). If the configuration memory programming has not completed properly, the FPGA will configure from the Golden Image stored in the external SPI memory (path shown in Figure 2). During this external memory configuration, the ASC I/O will be held in safe state. The I/O states of the FPGA and ASC are shown in Table 3. The FPGA Dual Boot operation is documented in detail in TN1204, [MachXO2 Programming and Configuration Usage Guide](#) and in TN1279, [MachXO3 Programming and Configuration Guide](#).

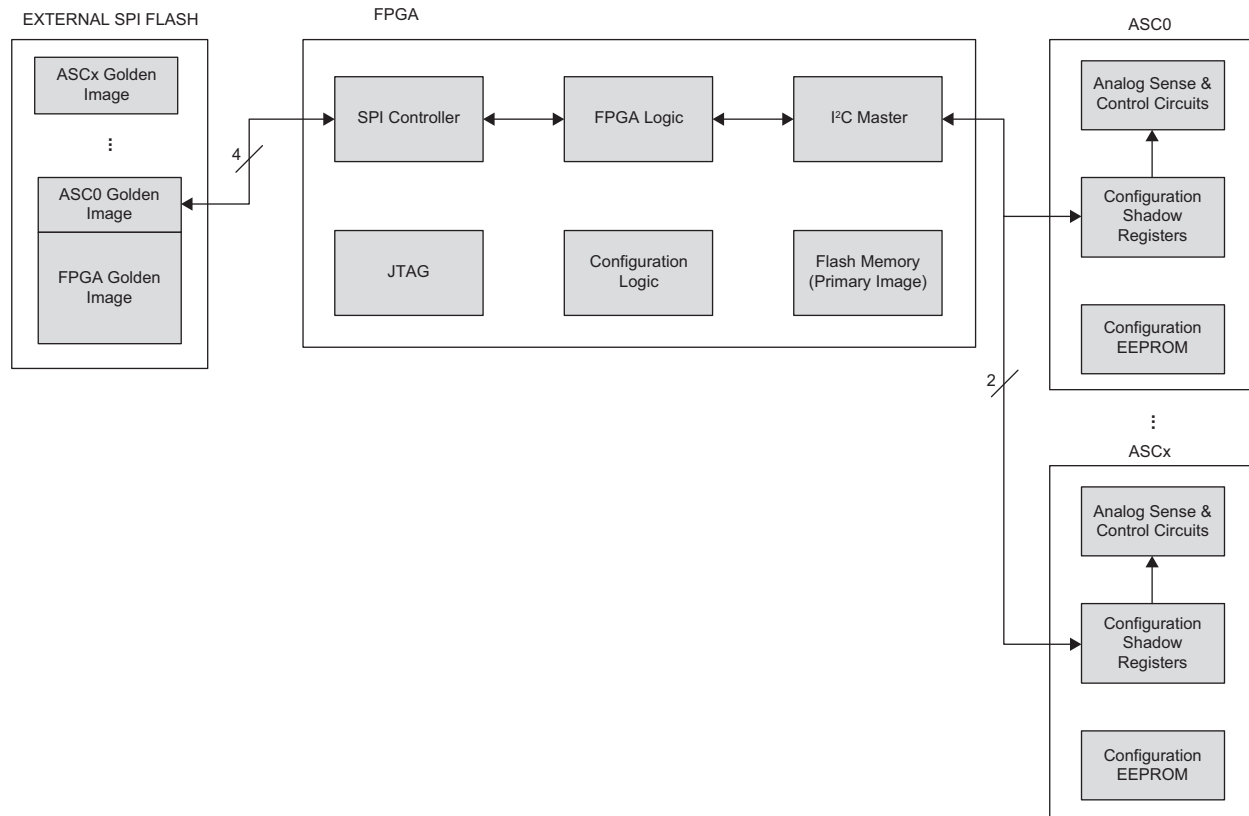
Figure 2. Dual Boot Stage 1 - FPGA Configuration from External SPI Flash



Stage 2. ASC Dual Boot

Stage 2 begins after the FPGA configuration has completed loading from the external SPI memory. The ASC Dual Boot IP block will now be operating in the FPGA logic. The IP block will read the ASC configuration image from the SPI memory, and program the corresponding ASC shadow registers over the I²C bus (path shown in Figure 3). The IP will perform a CRC check with the ASC at the conclusion of I²C programming. The IP will repeat the programming operation in case of a CRC fail. After a successful CRC calculation, the IP will proceed to program each of the additional ASC devices in the system. The IP will confirm if additional ASCs are mandatory or optional (as described in the [L-ASC10 Data Sheet](#)). If mandatory ASCs are not found on the I²C bus, the IP will hold the ASCs in safe state and wait for a Power-On-Reset. If optional ASCs are not found on the I²C bus, the IP will proceed to the next ASC in the system. The dual boot configuration method will update all ASC device settings except the I²C Write Protect feature and the UES codes. These features can only be updated through a successful EEPROM programming operation. The full configuration flow for the Platform Manager 2 Dual Boot is shown in the flow chart found in [Appendix A. Dual Boot Flow Chart](#).

Figure 3. Dual Boot Stage 2 - Configuration of ASC from External SPI Flash



I/O States During Dual Boot

The ASC and FPGA I/O have defined behavior during the Dual Boot process. These I/O states are shown in Table 2 below.

Table 2. I/O States

Pin Type	State during Dual Boot Stage 1 (FPGA)	State during Dual Boot Stage 2 (ASC)
FPGA PIO	Hi-Z	Logical Reset State
ASC GPIO1-6, GPIO10	Low / Active Pull-Down	Low / Active Pull-Down
ASC GPIO7-9	Hi-Z	Hi-Z
HVOUT	Low / Active Pull-Down	Low / Active Pull-Down
TRIM	Hi-Z	Hi-Z

The FPGA PIO states are dependent on the dual boot stage. During Dual Boot Stage 1 (FPGA dual boot), the FPGA PIO are held in their default configuration state of Hi-Z. Once the Dual Boot transitions to stage 2, the FPGA PIO will enter their reset state as defined by the configured design. FPGA PIO controlled by logic generated by Platform Designer will be held in their reset state until the completion of Dual Boot Stage 2. FPGA PIO controlled by user logic can be held in their reset state by monitoring the DB_BUSY signal generated by Platform Designer. This signal is logic 1 during Dual Boot Stage 2 and will be cleared to logic 0 when Stage 2 has completed. See the references section for more details on working with Platform Designer.

The ASC GPIO, HVOUT, and TRIM pins are held in their hardware defined safe state during both Stage 1 and Stage 2. See Table 2 for the safe state of each of these signals.

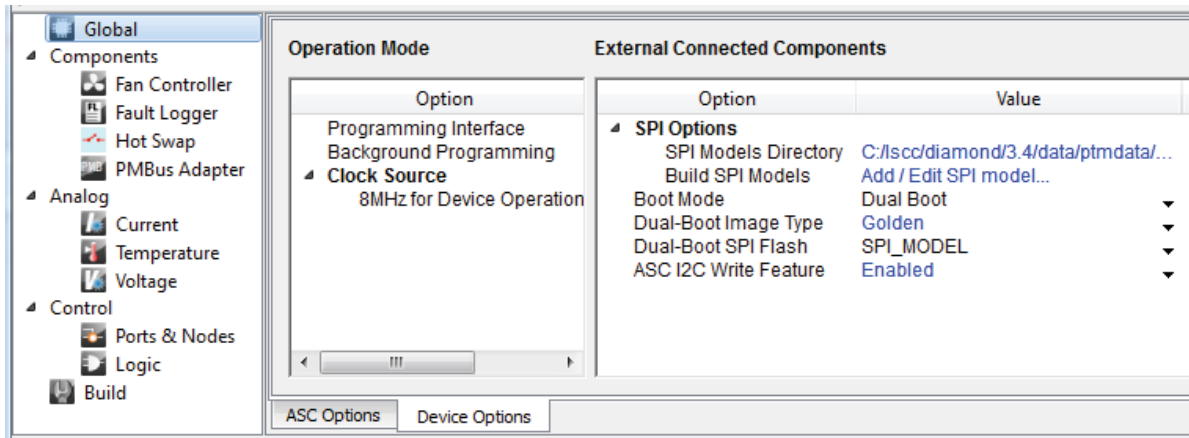
Configuring Dual Boot in Platform Designer

The Platform Designer tool (a part of Lattice Diamond software) is used to enable Dual Boot support in Platform Manager 2 designs. The Dual Boot setting is managed in the Global Configuration View, under the Device Options tab, in the External Connected Components section.

Boot Mode – This setting is used to enable Dual Boot support.

- Normal – This is the default project setting which disables Dual Boot Support.
- Dual Boot – Enables Dual Boot support in the project. If Dual Boot Mode is selected, two new settings appear – Dual Boot Image Type and Dual Boot SPI Flash, as shown in Figure 4 below:

Figure 4. Boot Mode Setting in Platform Designer

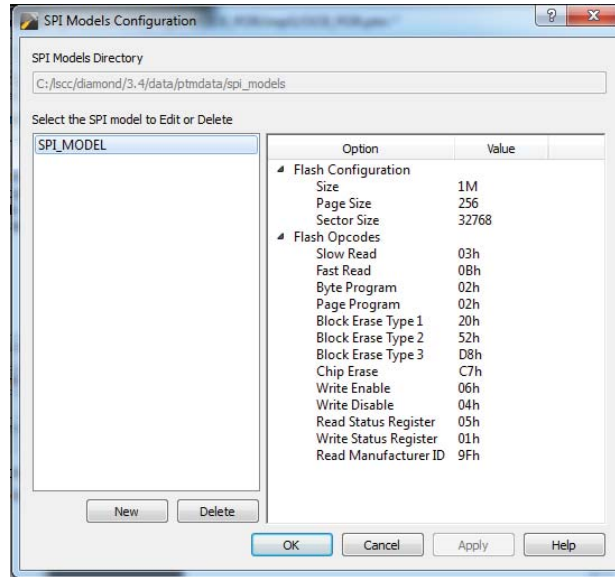


Dual Boot Image Type – This setting specifies the type of image which will be generated by Platform Designer.

- Golden – This setting is used to generate the Golden (backup) image stored in the external SPI flash. This image includes the Dual Boot logic used to reconfigure the ASC devices in case of a failure during programming. Platform Designer generates an .mcs file for programming the external SPI Flash when this setting is used.
- Primary – This setting is used to generate the image programmed into the internal FPGA Flash during production or for images used for background upgrades. Building a project with this setting automatically configures the FPGA to support Dual Boot mode, although this normal image will not include the Dual Boot logic. Platform Designer generates a .jed file and .hex files for programming the FPGA configuration Flash and ASC EEPROMs.

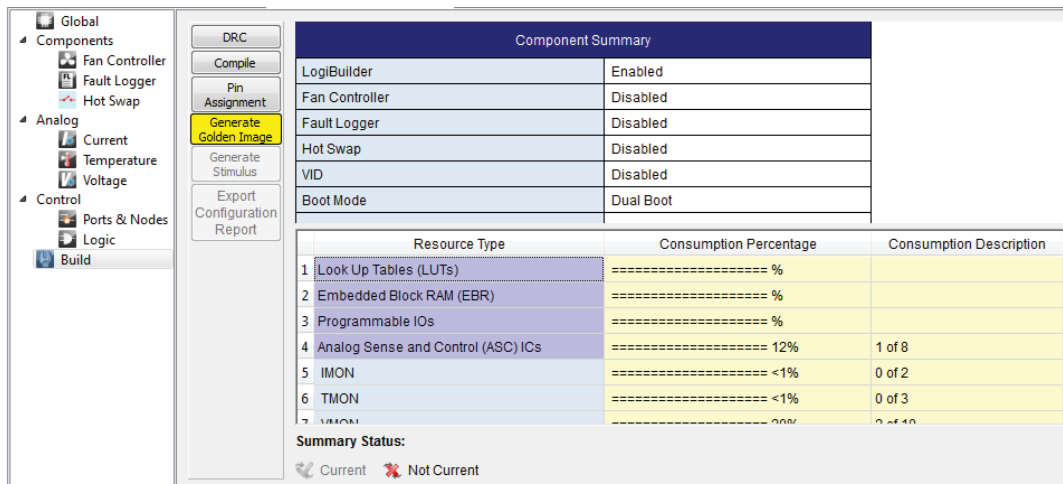
Dual Boot SPI Flash – This setting specifies the external SPI flash which stores the Golden image. The drop-down menu is populated with any previously defined SPI Flash models. Choosing **Add/Edit SPI Model** from the drop down menu brings up the SPI Models Configuration dialog box shown in Figure 5. The SPI model configuration should be defined based on the data sheet information of the SPI Flash used in the design. The chosen SPI memory must support the 0x03 Read Command, this is a requirement of the FPGA Dual Boot mechanism.

Figure 5. SPI Models Configuration



Once the boot mode settings are chosen, the rest of the analog and logic settings can be completed according to the normal Platform Designer flow. The Build view operations automatically update to generate either the golden image .mcs file or the primary image .jed file. The **Generate** button in the Build view updates to provide a visual cue that the Golden Image is being generated. This is shown in Figure 6. The primary and golden image projects can be managed as separate projects or separate implementations inside Lattice Diamond software.

Figure 6. Generate Golden Image Button in Build Tab



Programming Considerations for Reliable In-Field Updates

The Platform Manager 2 Dual Boot mode is designed to support reliable in-field updates. The Dual Boot mode is a backup mechanism that is most commonly activated due to programming failures. There are several considerations when programming systems using Dual Boot mode.

Programming Golden Image

The Golden Image is the backup image stored in the external SPI memory. This image includes the FPGA configuration image (including Dual Boot IP block) and the configuration images for the ASCs included in the system. Platform Designer is used to generate this combined image as described in the [Configuring Dual Boot in Platform Designer](#) section. The Golden Image memory map is dependent on the size of the FPGA in the system and the number of ASCs in the system. Figure 7 shows an example memory map using the MachXO2-7000 and 8 ASCs (the largest available combination in Platform Designer). The image start addresses in other device combinations are shown in Table 3.

Figure 7. Golden Image Memory Map - MachXO2-7000 and 8 ASCs

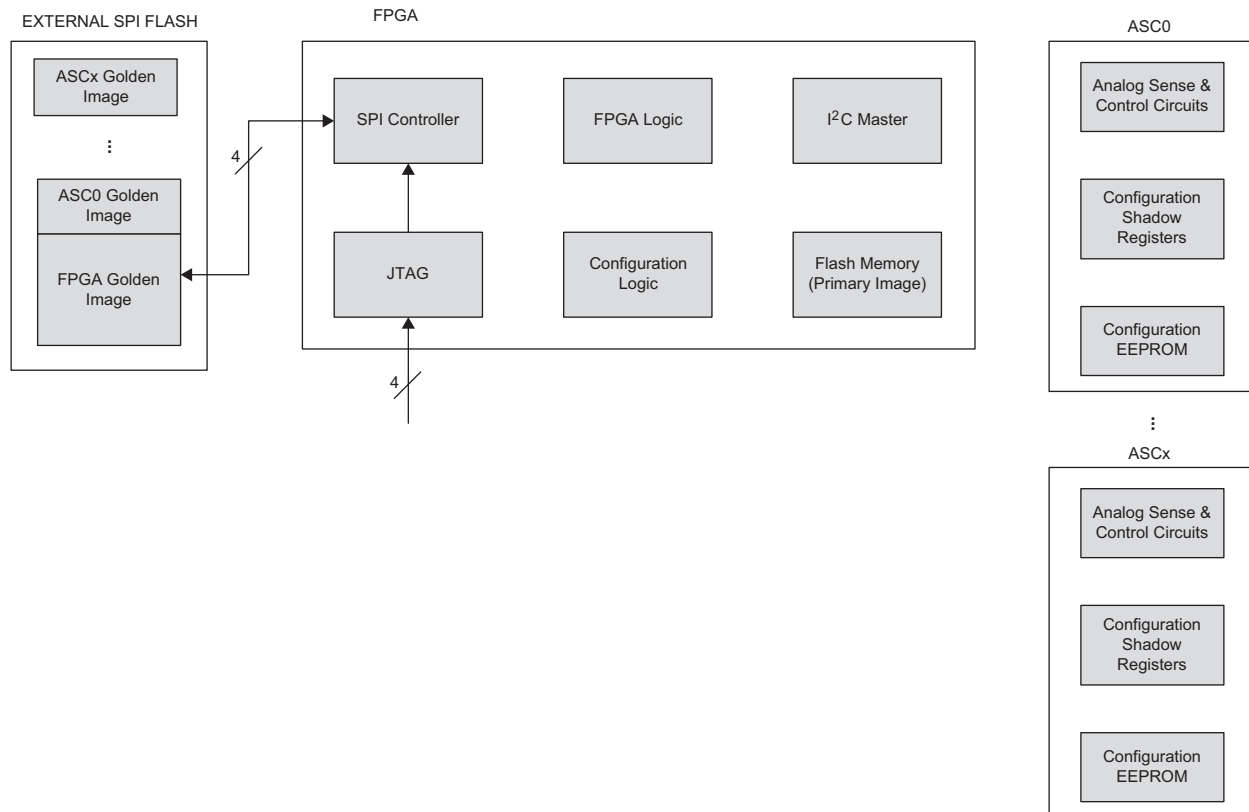
ASC7 Golden Image	03C800h
ASC6 Golden Image	03C700h
ASC5 Golden Image	03C600h
ASC4 Golden Image	03C500h
ASC3 Golden Image	03C400h
ASC2 Golden Image	03C300h
ASC1 Golden Image	03C200h
ASC0 Golden Image	03C100h
	03C000h
MachXO2 Golden Image	
	010000h
EMPTY	000000h

Table 3. Golden Image Memory Map - Start Addresses and Image Size

Golden Image Start	LPTM21 LPTM21L	MachXO2- 640	MachXO2- 1200	MachXO2- 2000	MachXO2- 4000	MachXO2- 7000
FPGA	010000h	010000h	010000h	010000h	010000h	010000h
ASC0	01A800h	015800h	01A800h	01F000h	029800h	03C000h
ASC1	01A900h	015900h	01A900h	01F100h	029900h	03C100h
ASC2	01AA00h		01AA00h	01F200h	029A00h	03C200h
ASC3	01AB00h		01AB00h	01F300h	029B00h	03C300h
ASC4				01F400h	029C00h	03C400h
ASC5				01F500h	029D00h	03C500h
ASC6					029E00h	03C600h
ASC7					029F00h	03C700h
Image End	01AC00h	015A00h	01AC00h	01F600h	02A000h	03C800h
Total Image Size (Mbits)	0.88	0.71	0.88	1.03	1.38	1.99

The external SPI memory can be programmed using the JTAG interface of the FPGA. The FPGA includes a hardware bridge between the JTAG and the SPI, as shown in Figure 8. The **SPI Flash Programming Mode** in Diamond Programmer is used to program the Golden Image. See the Diamond software user guide in the [References](#) section for more details.

Figure 8. Programming External SPI Flash with Golden Image



The external SPI Flash must be programmed with the Golden Image before primary image programming. The primary image must be programmed into the internal configuration Flash after the external SPI flash programming is complete.

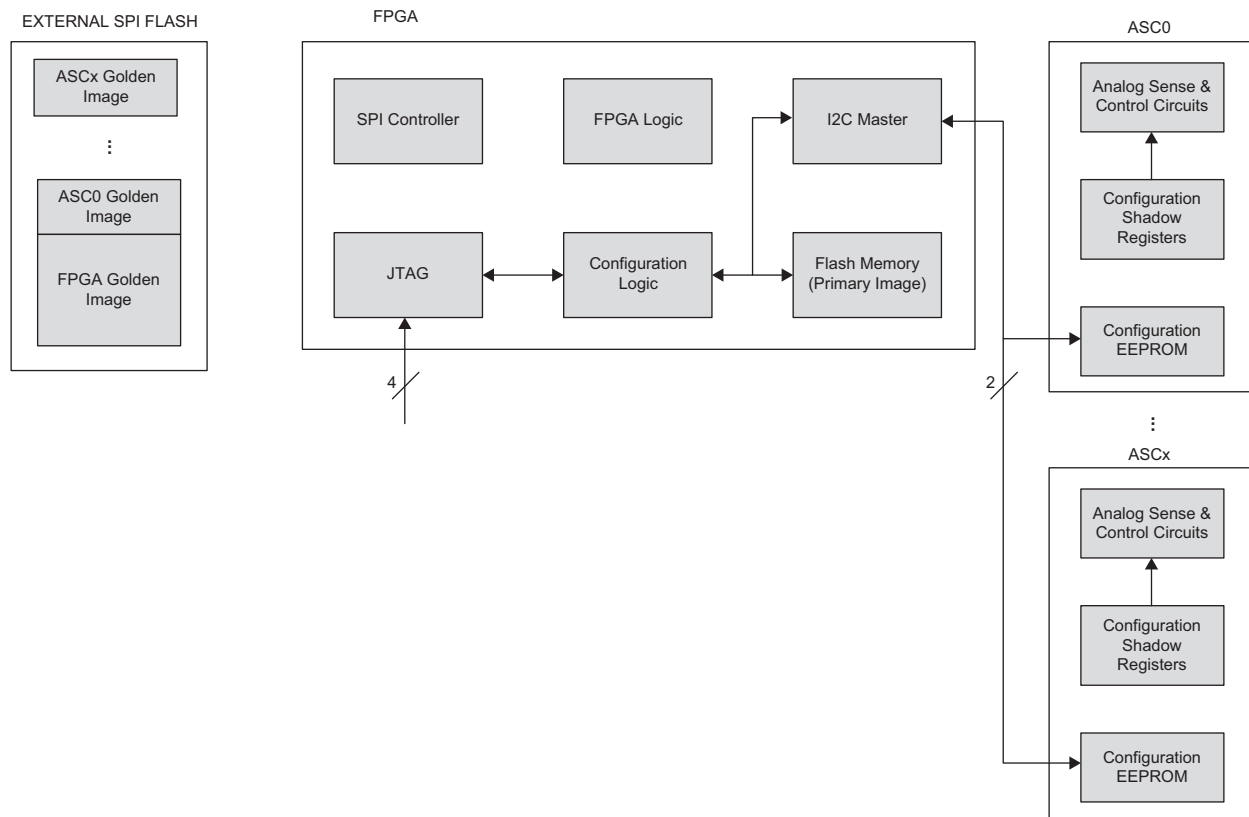
Background Programming of Primary Image

The primary image is the image stored in the internal Flash of the FPGA and the EEPROM in the ASC devices. During manufacturing, the primary image .jed and .hex files should be programmed after the external SPI memory has been programmed.

Platform Manager 2 supports background programming for reliable in-field updates. The background programming mechanism enables programming of the internal FPGA flash memory and ASC configuration EEPROM, without interrupting the device operation. After a successful background programming operation, the device loads the new configuration image after the next power cycle. Should there be an issue in the background programming, the Dual Boot mechanism takes over to safely configure Platform Manager 2 using the Golden Image in the external SPI Flash after the next power cycle.

Background programming is accomplished using the **PTM Background Programming** mode in Diamond Programmer (see the Diamond software user guide in the [References](#) section for more details). The Background Programming mechanism uses the hardware connections shown in Figure 9.

Figure 9. Background Programming of Platform Manager 2 and ASC



Demonstrating Dual Boot Operation

The Dual Boot operational design can be confirmed by executing a short series of steps using Diamond Programmer. The Primary and Golden images will both need to be built using the steps described in the Configuring Dual Boot in Platform Designer. (Note that using Primary and Golden images with external observational differences – such as different blinking LEDs – increases the clarity of the Dual Boot demonstration). The steps to demonstrate the Dual Boot are as follows:

1. **Program the External SPI-Flash using the Golden Image** – The first step is to perform an Erase-Program-Verify SPI Flash operation in Diamond Programmer using the Golden Image .mcs file. This stores the backup image into the external SPI Flash. At the conclusion of this step, the FPGA will be in Master-SPI configuration mode and will be held in configuration state until power-on-reset.
2. **Program the Internal Configuration Flash using the Primary Image** – The next step is to perform an Erase-Program-Verify PtM operation in Diamond Programmer using the Primary Image .jed file and Primary Image .hex file for each ASC. This stores the primary image into the internal FPGA Flash and the ASC configuration EEPROM. At the conclusion of this step, the FPGA will begin operating based on the internal Flash configuration and the ASC will be operating based on the configuration EEPROM.
3. **Background Erase the Internal Configuration Flash** – The PtM Background Erase operation can be used to simulate an incomplete background programming operation. This step erases the internal FPGA Flash. The operation should continue without interruption during the background erase step.
4. **Force a Power-On-Reset to the FPGA and ASC** – A power-on-reset causes the FPGA to refresh its configuration. Since the internal FPGA flash was erased in step 3, the FPGA will configure from the external SPI Flash (Stage 1 Dual Boot) and then configure the ASC (Stage 2 Dual Boot).
5. **Background Program the Internal Configuration Flash** – Execute a PtM Background Erase-Program-Verify operation using the Primary Image .jed file. This stores the primary image into the internal FPGA Flash and the ASC configuration EEPROM. The operation based on the Golden image will continue without interruption during this background step.
6. **Force a Power-On-Reset to the FPGA and ASC** – A power-on-reset causes the FPGA to refresh its configuration. Since the internal Flash and ASC EEPROM were programmed successfully in step 5, the FPGA will configure based on the internal FPGA Flash and the ASC will configure based on the EEPROM.

Hardware Connections Required for Dual Boot Mode

Table 4 lists the pins which are used for Dual Boot Mode in Platform Manager 2 designs. Figure 10 shows the required signal connections between devices using Dual Boot Mode. For additional details on these pins, see the data sheet links in the [References](#) section.

Table 4. Dual Boot Pin Descriptions

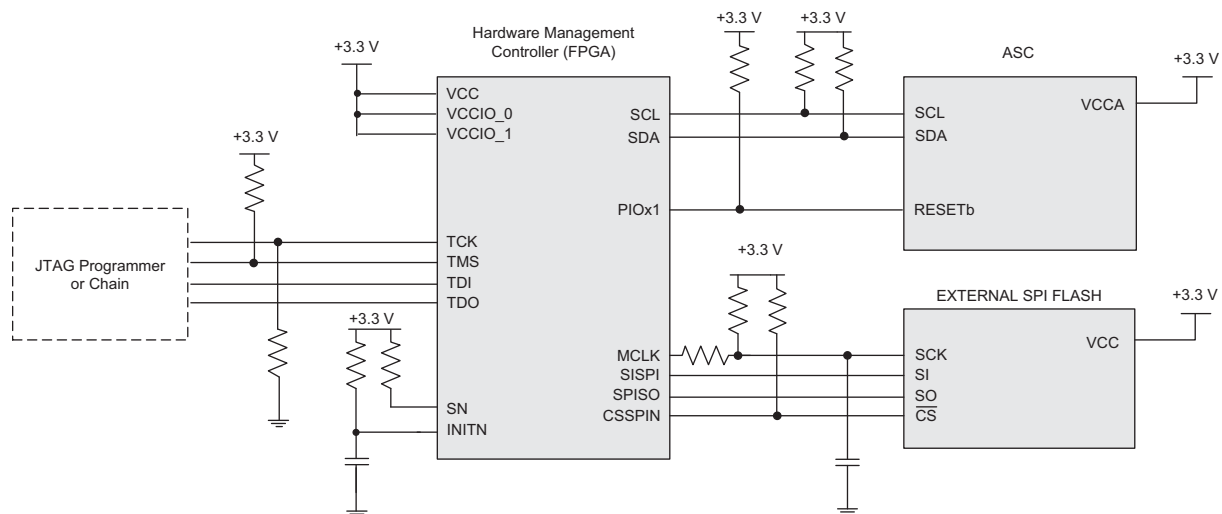
Pin Name	Function
MCLK	Clock output from the FPGA Configuration Logic and Master SPI controller. Connect MCLK to the SCLK input of the Slave SPI Memory. A 22 Ohm series resistor and a 1 k pull up resistor to VCC are recommended. Also, a 10 pF to 15 pF capacitor should be connected between the SPI clock pin and ground.
SISPI	Serial Data output from the FPGA to the slave SPI Memory SI input.
SPISO	Serial Data input to the FPGA configuration logic from the slave SPI memory SO output
CSSPIN	Chip select output from the FPGA configuration logic to the slave SPI memory holding configuration data for the FPGA. 10K external pull-up resistor to VCC is recommended.
SCL / SCL_M ^{1, 2, 3}	I2C Bus Clock. Connect to SCL of all ASC devices. External pull-up resistor required.
SDA / SDA_M ^{1, 2, 3}	I2C Bus Data. Connect to SDA of all ASC devices. External pull-up resistor required.
SN	Must have pull-up to enable I ² C after dual boot. 10K pull-up resistor recommended.
TCK	JTAG test clock, used for JTAG programming. 4.7K pull-down resistor recommended.
TMS	JTAG test mode select, used for JTAG programming. 4.7K pull-up resistor recommended.
TDI	JTAG test data input, used for JTAG programming.
TDO	JTAG test data output, used for JTAG programming.
RESETb (ASC Device)	ASC Reset input/output. Connect all mandatory RESETb together and to FPGA PIO, optional RESETb should be connected to individual FPGA PIO. A 10K pull-up to the ASC VCCA is required.
INITN	Open drain configuration pin. The FPGA pulls this pin low prior to beginning configuration. An RC circuit using 100K resistor and 10 uF capacitor is recommended to ensure proper configuration across VCC ramp rates.

1. LPTM21 device only

2. FPGA I²C pins may benefit from noise reduction techniques outlined in Lattice White Paper, [Improving Noise Immunity for Serial Interface](#).

3. ECP5 does not have dedicated I²C pins. The designer can assign any I/O pins for this function.

Figure 10. Hardware Connections Required for Dual Boot Mode



Recommended INITN Pin Handling

The INITN pin is an open-drain configuration pin. After POR, the INITN pin will be pulled low by the FPGA. The FPGA will release the INITN pin when it is ready to begin configuration. An internal pull-up will pull the signal high, and configuration will begin.

Adding an RC circuit to the INITN is recommended to delay the configuration start time. This will ensure that slower VCC ramp rates will not cause an issue during the FPGA dual boot. Without the RC circuit, the FPGA may begin the dual boot process of fetching its configuration image from the external SPI Flash before the SPI Flash POR has properly completed. An RC circuit using a 100 kOhm resistor (R265) and 10 uF capacitor (C102) is installed on the Platform Manager 2 evaluation board and delays the boot process for about 150 ms while the supply completes ramping. The figures below show the dual boot behavior with (Figure 11) and without (Figure 12) the RC circuit on INITN.

Figure 11. Dual Boot Complete - RC Circuit slows INITN

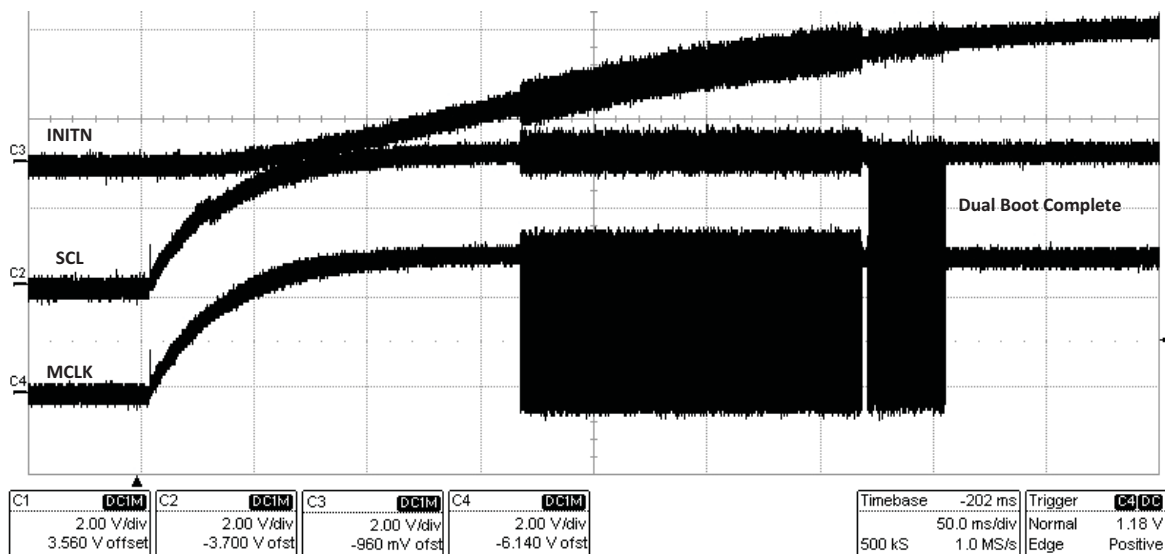


Figure 12. FPGA Dual Boot Incomplete - No RC Circuit on INITN

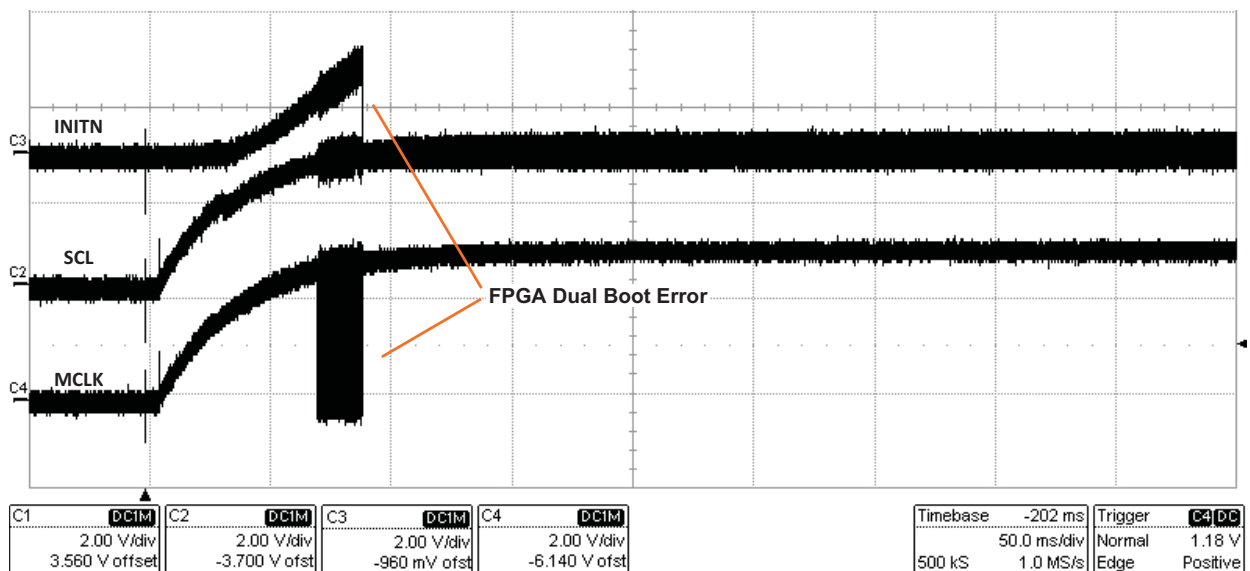


Figure 11 shows the INITN, SCL, and MCLK signals. The slow rising INITN delays the configuration start. MCLK begins to toggle, indicating the FPGA dual boot process, about 150 ms after POR. After the FPGA dual boot process completes, the SCL signal and MCLK signal toggle, indicating the ASC dual boot.

Figure 12 shows the same signals. In Figure 12, the INITN signal rises much faster due to the lack of RC circuit. The MCLK begins to toggle, indicating the beginning of the FPGA dual boot process. However the dual boot exits shortly after, with the INITN pin held low. This indicates a configuration failure due to a communication failure with the external SPI Flash right after POR.

The input supply in these examples rises from 0 to 3.3 V in about 75 ms. Supplies that take longer to stabilize may require a longer delay RC circuit.

The RC circuit will only delay the configuration start process if the INITN pin is setup as a configuration pin and not a general purpose I/O. This is configured in the FPGA Feature Row and this setting is automatically generated by Platform Designer when the Dual Boot – Primary image is built. For more details on the INITN pin function see TN1204, [MachXO2 Programming and Configuration Usage Guide](#).

Summary

The combination of Dual Boot Mode and background programming support ensure that Platform Manager 2 designs can be reliably updated in field. Platform Manager 2 designs can be built using several combinations of Lattice devices as listed in Table 5.

Table 5. Lattice Platform Manager 2 Design Options

Central Hardware Manager	Hardware Management Expander	Number of Expanders Supported
LPTM21	LPTM21L or L-ASC10	0 – 3
LPTM21L	LPTM21L or L-ASC10	0 – 3
MachXO2	LPTM21L or L-ASC10	1 – 8
MachXO3	LPTM21L or L-ASC10	1 – 8
ECP5	LPTM21L or L-ASC10	1 – 8

The operational details of Dual Boot and background programming have been described, along with the information needed to implement Dual Boot in your hardware management design.

References

- FPGA-DS-02038 (previously DS1042), [L-ASC10 Data Sheet](#)
- FPGA-DS-02036 (previously DS1043), [Platform Manager 2 Family Data Sheet](#)
- TN1204, [MachXO2 Programming and Configuration Usage Guide](#)
- TN1279, [MachXO3 Programming and Configuration Usage Guide](#)
- TN1225, [Platform Manager 2 Hardware Checklist](#)
- TN1216, [LatticeECP3, LatticeECP2/M, ECP5 and ECP5-5G Dual Boot and Multiple Boot Feature](#)
- White Paper, [Improving Noise Immunity for Serial Interface](#)
- [Platform Designer 3.1 User Guide](#)

Technical Support Assistance

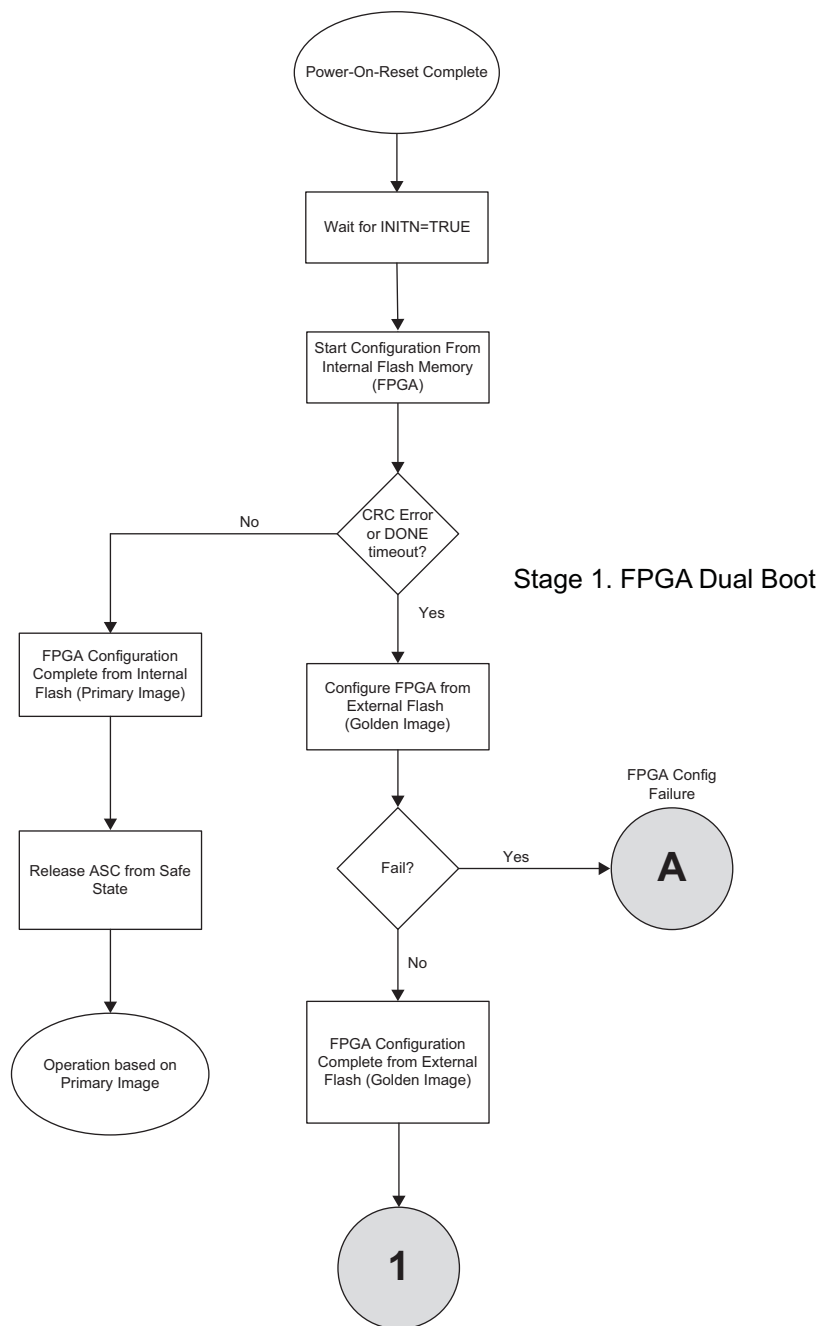
For assistance, submit a technical support case at www.latticesemi.com/techsupport.

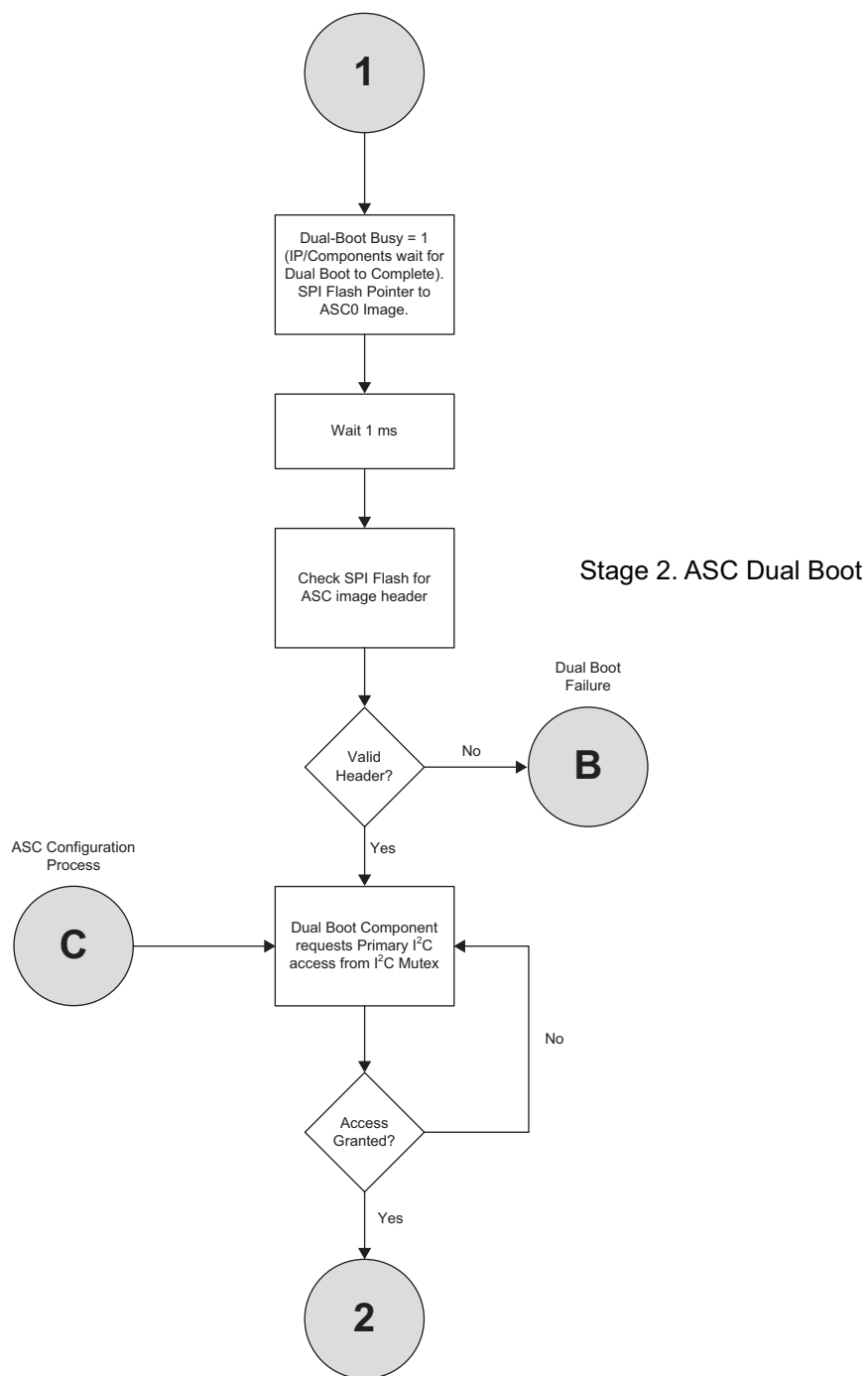
Revision History

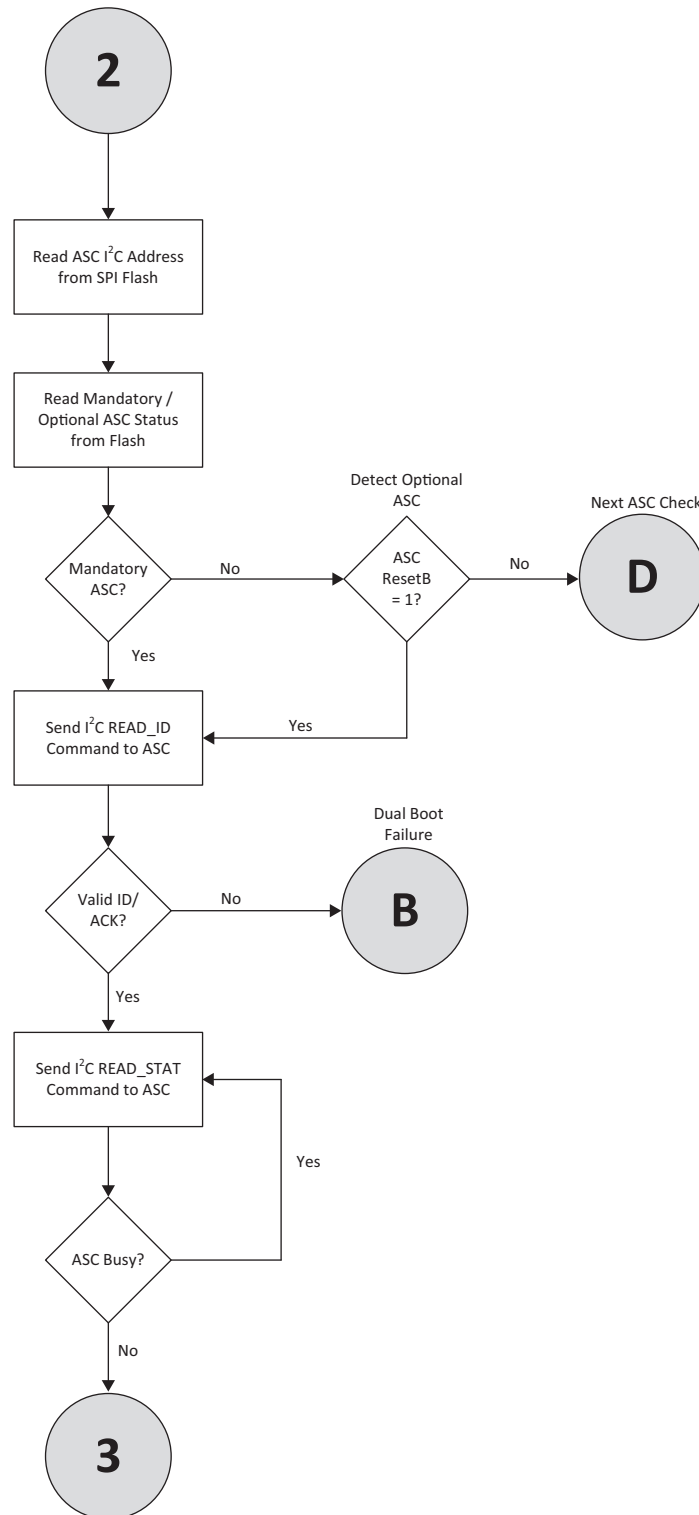
Date	Version	Change Summary
October 2019	1.3	Changed document number from TN1284 to FPGA-TN-02078.
		Added reference to LPTM21L, MachXO3, and ECP5.
		Updated Introduction section. Added Table 1.
		Updated Hardware Connections Required for Dual Boot Mode section. Revised Table 4 and Figure 10.
		Updated Summary section. Added Table 5.
		Updated References section. Added TN1279, MachXO3 Programming and Configuration Usage Guide. Indicated new document number of Platform Manager 2 Family Data Sheet.
		Updated Technical Support Assistance .
		Added Disclaimers section.
March 2015	1.2	Updated Stage 2. ASC Dual Boot section. Added information on UES code.
		Updated Configuring Dual Boot in Platform Designer section. Updated Figure 4, Boot Mode Setting in Platform Designer and Figure 5, SPI Models Configuration to generic SPI Model.
		Updated Appendix A. Dual Boot Flow Chart. Corrected error in flow chart.
August 2014	1.1	Updated Introduction section. Added Reset Distribution as Control Plane function.
		Re-organized and updated Theory of Operation section. Updated Figure 1, Configuration from Internal Memory (After Successful Programming)
		Added I/O States During Dual Boot section.
		Re-organized Configuring Dual Boot in Platform Designer section.
		Added Demonstrating Dual Boot Operation section.
		Updated Hardware Connections Required for Dual Boot Mode section. Updated Table 3, Dual Boot Pin Descriptions. Added footnote.
		Added Figure 10, Hardware Connections Required for Dual Boot Mode. Added Recommended INITN Pin Handling section.
		Added reference to White Paper, Improving Noise Immunity for Serial Interface.
February 2014	01.0	Updated Appendix A. Dual Boot Flow Chart. Updated figures. Identified Stage 1 - FPGA Dual Boot and Stage 2 - ASC Dual Boot.
		Initial release.

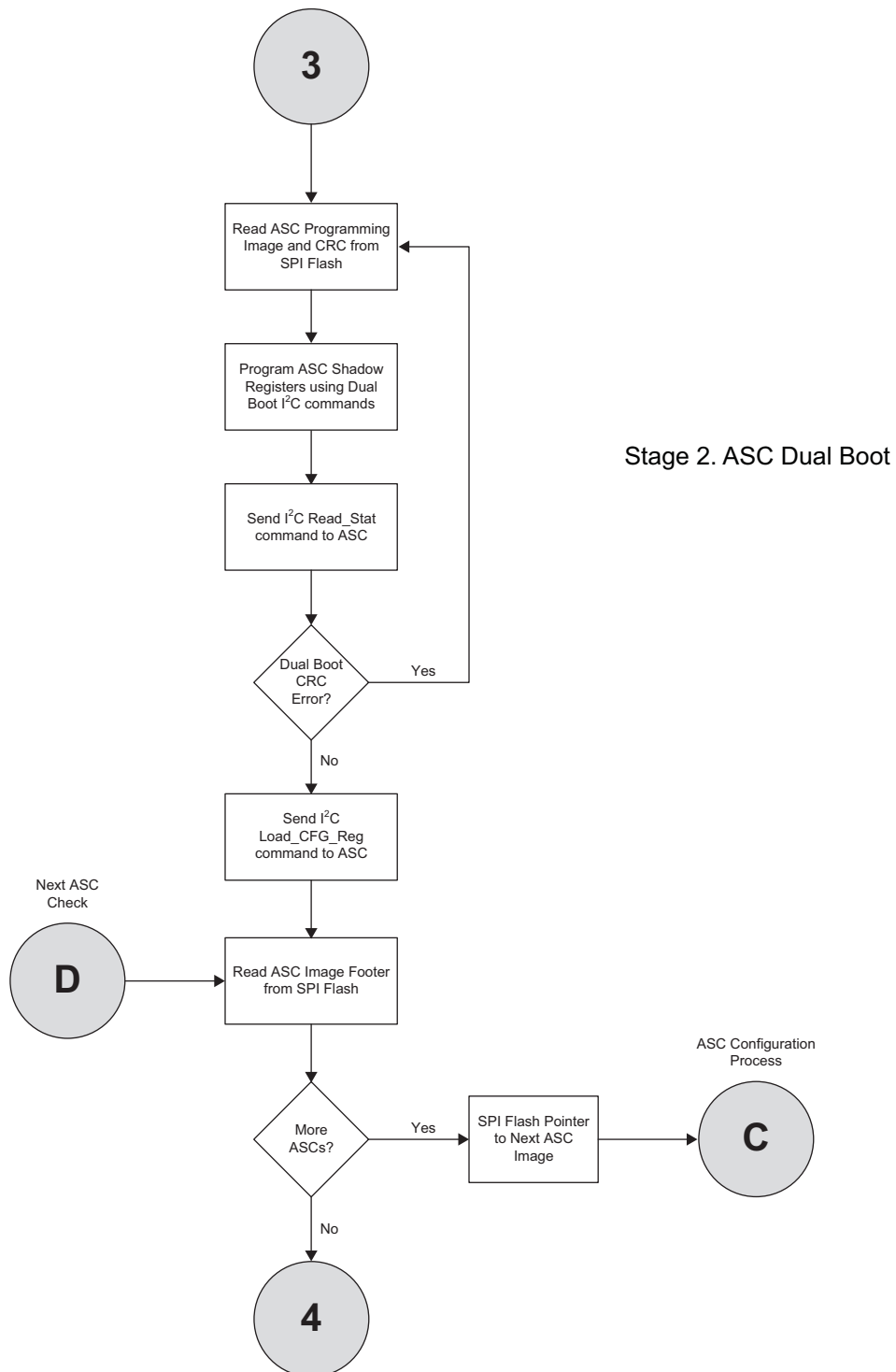
Appendix A. Dual Boot Flow Chart

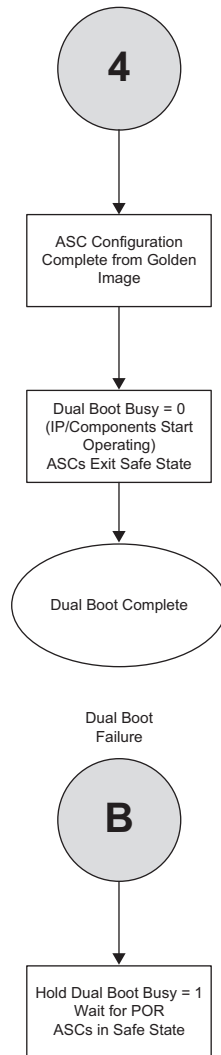
Figure 13. Platform Manager 2 Dual Boot Flow











Disclaimers

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