

RT PolarFire Device Family Enhancements Compared to PolarFire Devices Application Note

AN4903



Introduction [\(Ask a Question\)](#)

The Radiation-Tolerant (RT) versions of the PolarFire® FPGA and SoC devices are based on the non-volatile and reprogrammable PolarFire device family from Microchip, which are manufactured using 28 nm technology. This article provides an overview of the differences between the RTPF500T and RTPF500ZT device versions. Similarly, a comparison between the RT PolarFire SoC RTPFS###ZT devices and the PolarFire SoC MPFS###T devices is provided. For more information about these products and user documentation, see the [RT PolarFire](#) and [RT PolarFire SoC](#) web pages.

Document Status

The current status of this document is:

- RTPF500T: Production
- RTPF500ZT: Production
- RTPFS460ZT and RTPFS160ZT: Advance

There are three levels in the document status:

Advance	Initial estimated information from simulations.
Preliminary	Information based on simulation and/or initial characterization.
Production	Final production data.



Important: RTPF500T is not recommended for new designs. RTPF500T is being replaced by RTPF500ZT FPGAs.

For information about the errata associated with RTPF500T, RTPF500ZT, RTPFS460ZT and RTPFS160ZT devices, see [RT PolarFire FPGA and SoC FPGA Devices Errata](#).

Table of Contents

Introduction.....	1
1. High-Level Differences.....	3
2. System Controller Suspend Mode (SCSM).....	6
2.1. Processing MSS Reboot Requests During SCSM.....	7
3. System Services.....	8
3.1. Requesting System Controller Services Using PFSOC_SCSM (RTPFS460ZT and RTPFS160ZT Only).....	9
3.2. In-Flight Reprogramming.....	10
4. Radiation Tolerance.....	12
5. Device Compatibility Between RTPF500T and RTPF500ZT.....	14
5.1. Package Compatibility.....	14
5.2. Bitstream Compatibility.....	14
5.3. Design Migration from RTPF500T to RTPF500ZT.....	14
6. Device Compatibility Between MPFS###T and RTPFS###ZT Devices.....	15
6.1. Package Compatibility.....	15
6.2. BitStream Compatibility.....	15
7. I/O Bank Voltages.....	16
8. Appendix 1: Device Migration Process from RTPF500T to RTPF500ZT Using Libero SoC.....	17
9. Conclusion.....	26
10. Revision History.....	27
Microchip FPGA Support.....	29
Microchip Information.....	30
Trademarks.....	30
Legal Notice.....	30
Microchip Devices Code Protection Feature.....	30

1. High-Level Differences (Ask a Question)

The standard device offerings for Transceivers (T), Data Security (S), Speed Grades (STD and -1), Lower Static Power (L) and screening levels are listed in [RT PolarFire Product Overview](#). In addition to the standard device offerings, RTPF500T and RTPF500ZT are the two primary RT PolarFire versions also offered. The following table lists the high-level differences between RTPF500T and RTPF500ZT versions.

Table 1-1. High-Level Differences Between RTPF500T and RTPF500ZT

RTPF500T	RTPF500ZT
Uses the same silicon as commercial PolarFire® MPF500T FPGA, but with modified flip-chip die bump spacing to enable hermetically-sealed ceramic packaging and space-grade device screening flows.	Silicon design has been enhanced over RTPF500T to facilitate the added features and improved performance described in this column.
User design access to system services requiring the internal system controller are unavailable when using System Controller Suspend Mode (SCSM) ¹ .	Supports on-demand access to system controller services after SCSM exit during normal operation.
Only supports JTAG and SPI-target in-flight reprogramming modes.	Supports JTAG, SPI-target, In-Application Programming (IAP) and Auto-Update in-flight reprogramming modes.
Power-loss or brownout of VDD, VDD18, or VDD25 supplies during an sNVM page write can corrupt the page, leaving it as a read-only page, preventing further page writes during design operation. Restoring write access to the corrupted page requires reprogramming the device using a bitstream containing content for the affected sNVM page. For more information, see PolarFire System Services User Guide , Rev F or later.	Enhanced system controller sNVM write service to prevent sNVM pages from becoming read-only in the event of a power loss or brownout during a page write.
Baseline PolarFire FPGA radiation tolerance, as documented in published test reports, and summarized in Radiation Tolerance .	Enhanced radiation tolerance over RTPF500T, as summarized in Radiation Tolerance .
No ECC on internal sNVM/pNVM data storage areas ² .	Added SECCDED ECC to internal sNVM/pNVM data storage areas to mitigate heavy-ion NVM bit errors ² .
sNVM organized into 221 pages, each of which holds 252 bytes of non-authenticated plaintext or 236 bytes of authenticated plaintext or ciphertext.	sNVM organized into 221 pages, each of which holds 220 bytes of non-authenticated plaintext or 204 bytes of authenticated plaintext or ciphertext. The user data storage capacity per page is reduced due to the overhead required to implement SECCDED ECC on sNVM data.
Transceiver VDDA supply supports a voltage of 1.0V or 1.05V. A 1.0V VDDA supports lane rates less than or equal to 10.3125 Gbps. A 1.05V VDDA can be used for all lane rates, and must be used for rates greater than 10.3125 Gbps. Must also use -1 speed grade above 10.3125 Gbps.	Transceiver VDDA supply must be 1.05V for all lane rates. Must use -1 speed grade above 10.3125 Gbps.

Notes:

- For more information about which device features are accessible on RTPF500T, when SCSM is enabled, see [PolarFire Family System Services User Guide](#).
- For more information, see the "pNVM/boot up failure Results" section in the [RT PolarFire RTPF500ZT Heavy Ion Test Results](#) report from LBNL, on February 6th, 2023.

The following table lists the high-level differences between PolarFire SoC MPFS####T devices and RT PolarFire SoC RTPFS####ZT devices.

Table 1-2. Differences Between MPFS####T and RTPFS####ZT

MPFS160T, MPFS460T	RTPFS160ZT, RTPFS460ZT
Terrestrial PolarFire® SoC devices	Silicon design has been enhanced by incorporating the changes made to RTPF500ZT, as outlined in the preceding table, and described in this column.

Table 1-2. Differences Between MPFS####T and RTPFS####ZT (continued)

MPFS160T, MPFS460T	RTPFS160ZT, RTPFS460ZT
Non-hermetic, COTS plastic ball grid array packaging, per PolarFire SoC Product Overview . Commercial substrate, capacitors, and lid. MIL-Temp device and package combinations use ruggedized vented packages.	RTPFS460ZT uses modified flip-chip die bump spacing to enable hermetically sealed, ceramic column CG1509 package. RTPFS460ZT is also offered in RT plastic, ball grid FC(G)1509. RTPFS160ZT is only available in RT plastic, ball grid FCV(G)784. RT plastic packages use RT substrate, RT capacitors, and hi-reliability lid. Plastic packages are not hermetically sealed. MPFS160T and RTPFS160ZT in FCV(G)784 are pin compatible. For more information, see RT PolarFire SoC Product Overview .
Transceiver VDDA supply supports a voltage of 1.0V or 1.05V. A 1.0V VDDA supports lane rates less than or equal to 10.3125 Gbps. A 1.05V VDDA can be used for all lane rates and must be used for rates greater than 10.3125 Gbps. Must also use -1 speed grade above 10.3125 Gbps.	Transceiver VDDA supply must be 1.05V for all lane rates. You must use -1 speed grade above 10.3125 Gbps.
Supports JTAG, SPI-target, In-Application Programming (IAP), and Auto-Update reprogramming modes for terrestrial applications.	Supports JTAG, SPI-target and enhanced In-Application Programming (IAP) and Auto-Update in-flight programming modes, per In-Flight Reprogramming .
Power-loss or brownout of VDD, VDD18, or VDD25 supplies during an sNVM page write can corrupt the page, leaving it as a read-only page, preventing further page writes during design operation. Restoring write access to the corrupted page requires reprogramming the device using a bitstream containing content for the affected sNVM page. For more information, see PolarFire Family System Services User Guide , Rev F or later.	Enhanced system controller sNVM write service to prevent sNVM pages from becoming read-only in the event of a power loss or brownout during a page write.
User design access to system services requiring the internal system controller is unavailable when using System Controller Suspend Mode (SCSM) ¹ .	On-demand access to system controller services is available after SCSM exit during normal operation.
During SCSM, MSS soft reset (that is, reboot) can be processed by directly connecting MSS REBOOT_REQUESTED_M2F output to PFSOC_SCSM macro's SC_WAKE input. No other connections to the SC_WAKE input are allowed.	During SCSM, the same approach is used to process MSS reset requests. Additionally, the PFSOC_SCSM SC_WAKE input can be driven by a dynamic user signal from the FPGA fabric, to temporarily exit suspend mode. See System Controller Suspend Mode (SCSM) and System Services .
Same baseline radiation tolerance as PolarFire and PolarFire SoC, including the RT PolarFire RTPF500T as documented in published test reports, and summarized in Radiation Tolerance .	Incorporates the radiation tolerance enhancements made to the RTPF500ZT, as summarized in the Radiation Tolerance .
No ECC on internal sNVM/pNVM data storage areas. For more information on upset rate, see Radiation Tolerance and ² .	Added SECDED ECC to internal sNVM/pNVM data storage areas to mitigate heavy-ion NVM bit errors ² .
sNVM organized into 221 pages, each of which holds 252 bytes of non-authenticated plaintext or 236 bytes of authenticated plaintext or ciphertext.	sNVM organized into 221 pages, each of which holds 220 bytes of non-authenticated plaintext or 204 bytes of authenticated plaintext or ciphertext. The user data storage capacity per page is reduced due to the overhead required to implement SECDED ECC on sNVM data.
PolarFire SoC Microcontroller Subsystem (MSS)	Same MSS design. Additional reference design for radiation-mitigated secure boot mode 2, using initial bootloader in ECC protected sNVM. User boot code is encoded with software-defined ECC and stored in eNVM. The sNVM bootloader decodes the user eNVM boot code, copies it to ECC protected L2 scratchpad, and executes the user boot. Application note and reference design to be published.
PolarFire SoC 128 KB eNVM without ECC	Same eNVM. Software defined SECDED ECC can be applied as mentioned. When using soft-ECC, eNVM usable space is reduced from 128 KB to 102 KB.

Notes:

1. For more information about which device features are accessible on PolarFire SoC MPFS devices when SCSM is enabled, see [PolarFire Family System Services User Guide](#).
2. For more information, see the "pNVM/boot up failure Results" section in [RT PolarFire RTPF500ZT Heavy Ion Test Results](#) report from LBNL, on February 6th, 2023.

2. System Controller Suspend Mode (SCSM) [\(Ask a Question\)](#)

For high-reliability applications, RT PolarFire FPGAs and SoCs can be configured to hold the internal system controller in reset after the device power-up process is complete. This is a default option for RT PolarFire FPGAs and SoCs in the Libero[®] SoC design suite. The SCSM ensures that the system controller is held in reset using a Single Event Upset (SEU) immune, self-refreshing, and TMR latch to prevent Single Event Effects (SEEs) from causing unintended system controller operations such as device programming or zeroization.

It is recommended to minimize the amount of time the internal system controller is active during flight because it is not protected against SEUs. Once SCSM is active, all system controller outputs to the FPGA fabric design go to logic "0". Therefore, it is important to configure the PF_INIT_MONITOR and PFSOC_INIT_MONITOR IP cores to Latch system controller outputs during SCSM, especially if the output signals are used to derive a Reset signal for the user logic.

When PolarFire (MPF), PolarFire SoC (MPFS) and RTPF500T devices are programmed with a bitstream that enables SCSM, the system controller is held in reset upon completion of the power-up process, assuming that the JTAG TRSTB pin is at a logic-low level. To exit SCSM during normal operation, the JTAG TRSTB input must be driven High. After driving TRSTB high, the system controller can be used for device programming until TRSTB returns to logic-low. SmartDebug can also be accessed while JTAG TRSTB is driven high.


However, for PolarFire (MPF), PolarFire SoC (MPFS), and RTPF500T devices using SCSM, certain system services that require the system controller to be active during normal design operation are not available after the device completes the power-up process. Nevertheless, you can still reprogram the device using one of the supported programming modes if the JTAG TRSTB pin is driven high externally to release the system controller from reset.

On MPF, MPFS, and RTPF500T devices, when SCSM is exited, the latches used by the PF_INIT_MONITOR or PFSOC_INIT_MONITOR IP core to latch the system controller outputs are cleared, causing any active-low fabric resets derived from output signals such as DEVICE_INIT_DONE to get asserted. Therefore, SCSM exit on these devices must only be performed prior to reprogramming the device, and you can plan the system to accommodate the fabric design entering its reset state before reprogramming. Once the device is successfully reprogrammed with a bitstream that continues to use SCSM, you must return the JTAG TRSTB pin to a logic-low state to re-enter SCSM.

In contrast, when using RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices, you can use the JTAG TRSTB signal to enter and exit SCSM during normal design operation. To exit SCSM and release the system controller from reset, drive JTAG TRSTB High. To re-enter SCSM and immediately put the system controller into reset, set JTAG TRSTB low. Even though you still need to configure the PF_INIT_MONITOR or PFSOC_INIT_MONITOR IP core to latch the system controller outputs when entering SCSM on these devices, the latched values are no longer cleared when SCSM is exited during normal operation. Additionally, during design operation, temporarily exiting SCSM enables user access to selected system controller services. In general, it is recommended to minimize the amount of time the system controller is active during flight to reduce the risk of SEUs.

Finally, RTPFS460ZT and RTPFS160ZT devices support an additional method to temporarily exit SCSM during normal operation. For these devices, instantiating the PFSOC_SCSM macro serves a dual-purpose:

1. It provides a means to process an MSS reboot request when operating with SCSM enabled, as described in the [Processing MSS Reboot Requests During SCSM](#). This use-model is also available on terrestrial PolarFire SoC MPFS devices.
2. For RTPFS460ZT and RTPFS160ZT devices, it also allows user software or firmware applications running on the MSS to exit SCSM on-demand for the purposes of running a device system service, as described in the following sections.

 **Important:** To avoid incomplete transactions, make sure there are no active Dynamic Reconfiguration Interface (DRI) transactions in progress before exiting SCSM on RTPF500ZT, RTPF5460ZT, and RTPF5160ZT devices.

Related Links


[RT PolarFire Programming User Guide](#)

[PolarFire FPGAs for Safety-Critical Applications](#)

2.1. Processing MSS Reboot Requests During SCSM (Ask a Question)

PolarFire SoC (MPFS###T) and RT PolarFire SoC (RTPFS###ZT) devices, when operating with System Controller Suspend Mode (SCSM) enabled, support temporarily waking up the internal system controller to process an MSS reboot request. The sources of an MSS reboot request include:

- Writing to specific registers in the MSS register map to request a reset, such as the MSS_RESET_CR.
- Assertion of specific MSS reset input signals, such as MSS_RESET_N_F2M.
- A watchdog timeout from WDOG0 connected to the E51 Monitor Core.
- A reset triggered by a CPU debugger.

 **Important:** To use a debugger, the JTAG TRST_N must be driven High to wake the system controller and thus the device would not be in SCSM during a debugger-initiated MSS reset.

This is accomplished by incorporating the following into the user design:

- In the MSS Configurator, navigate to the **Misc** tab and select "Expose Feedback ports to Fabric" to make the REBOOT_REQUESTED_M2F output port visible on the MSS component.
- From the IP Catalog Macro Library, instantiate the PFSOC_SCSM primitive macro.
- Connect the MSS output REBOOT_REQUESTED_M2F to the SC_WAKE input of the PFSOC_SCSM.
- Instantiate the SC_STATUS primitive macro and monitor the SUSPEND_EN output to confirm whether the system controller is suspended.
- This connection is illustrated for the PolarFire SoC MPFS devices in the "PolarFire SoC and RT PolarFire SoC Reboot" section in the [PolarFire Family Power-Up and Resets User Guide](#).

3. System Services [\(Ask a Question\)](#)

The internal system controller provides a variety of system services as described in the [PolarFire Family System Services User Guide](#).

MPF, MPFS, and RTPF500T devices that are configured to use SCSM have limited access to system services. The services available during SCSM include those that do not require the system controller to be active during normal design operation. These include access to: user crypto co-processor, POR digest check, tamper IO_Disable, user voltage detectors, UJTAG security monitor, and TVS. If the tamper macro is used, it must be configured to latch the following outputs before the device enters SCSM:

- Reset reason output
- Tamper flag [13] output for POR digest check

In contrast, the ability to temporarily exit SCSM to perform system services during normal operation is a useful addition to the RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices. It allows additional access to commonly used services such as:

- sNVM access
- Bitstream authentication service for updated bitstreams stored in the external SPI memory
- In-Application Programming (IAP) where the RT PolarFire controls an external SPI Flash memory containing the target bitstream.
- Auto-Update programming service using previously authenticated bitstreams
- Data security services including digital signatures, PUF emulation and Nonce generation.
- Device and design information
- SPI flash memory read service to copy data from an external SPI flash to MSS memory (SoC devices only).

Notes:

- For RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices, when exiting SCSM by driving the JTAG TRSTB pin high, it is required to wait 0.5 μ s after driving TRSTB high before requesting a system service.
- For RTPFS460ZT and RTPFS160ZT devices, when exiting SCSM by driving the SC_WAKE input high, it is required to wait 0.5 μ s after driving SC_WAKE high before requesting a system service.
- If a system service is requested while the device is operating in SCSM mode, it will be ignored. To confirm the system controller state before issuing the system service request, the user design must monitor the SUSPEND_EN output of the SC_STATUS macro. For more information about the SC_STATUS macro, see [PolarFire Macro Library Guide](#).
- The user design must schedule the re-entry into SCSM by waiting for any initiated system service to complete before re-entering SCSM. If temporary SCSM exit was performed using JTAG TRSTB, then SCSM re-entry occurs immediately after TRSTB is returned to a low state. For the RTPFS460ZT and RTPFS160ZT, if SCSM exit was performed using the PFSOC_SCSM SC_WAKE input, then SCSM re-entry occurs immediately after SC_WAKE is returned to logic low.
- After exiting SCSM on RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices during normal operation, the system service to perform a fabric digest check is not supported. This service powers-off the fabric and the fabric RAM block contents are not preserved. Furthermore, the device cannot re-enter SCSM until a subsequent device power-cycle or DEVRST_N assertion and release.
- After exiting SCSM on RTPF500ZT during normal operation, the Dynamic Reconfiguration Interface (DRI) is not available for I/O related DRI read or write operations. If needed, the user must perform I/O related DRI transactions while SCSM is active. This limitation does not apply to the RTPFS460ZT and RTPFS160ZT devices.

3.1. Requesting System Controller Services Using PFSOC_SCSM (RTPFS460ZT and RTPFS160ZT Only) [\(Ask a Question\)](#)

An enhancement added to the RT PolarFire SoC (RTPFS460ZT and RTPFS160ZT) over the terrestrial PolarFire SoC MPFS devices is the ability to drive the PFSOC_SCSM macro's SC_WAKE input High. This is achieved by using a dynamic user logic signal, to exit SCSM and keep the system controller active, until the signal returns to logic Low. This feature enables a user design with SCSM enabled to access system controller services during normal operation by following these example steps:

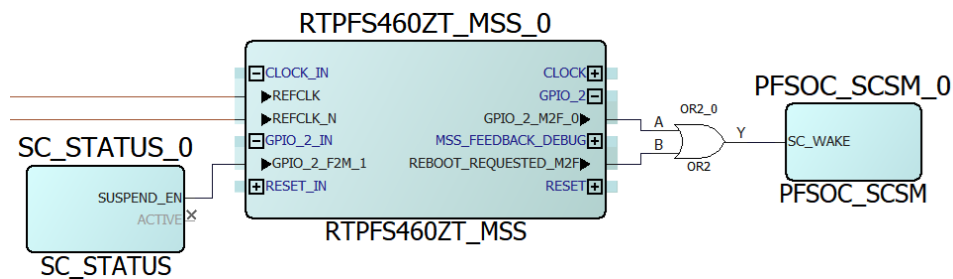
- The user application writes a logic-1 to an MSS GPIO output to the fabric or to a memory-mapped fabric register connected to the PFSOC_SCSM macro's SC_WAKE input to exit the SCSM.
- The application reads the SC_STATUS macro's SUSPEND_EN output to confirm that the system controller is out of suspend mode.
- The application requests a system service that requires processing by the system controller. This can be done using the mss_sys_services firmware drivers.
- Upon completion of all requested system controller services, the application writes a logic-0 to SC_WAKE to re-enter suspend mode and confirms successful re-entry by monitoring the SUSPEND_EN output.
- Similarly, user logic in the fabric can initiate and manage the SCSM exit and re-entry by driving the SC_WAKE input and monitoring the SUSPEND_EN output, along with the PF_SYSTEM_SERVICES core from the Libero SoC IP catalog. Fabric logic could also include a soft Mi-V processor that either connects to the PF_SYSTEM_SERVICES core using firmware drivers from the Libero SoC Firmware Catalog, or uses the MSS FIC fabric initiator port to access the MSS register-mapped system services mailbox using MSS system services firmware drivers. For more information, see ³.



Important:

1. This feature requires Libero SoC v2025.1 or later. Using an earlier version of Libero SoC results in a Compile error when fabric logic drives the SC_WAKE input, because older versions enforce the same SC_WAKE connectivity rules as PolarFire SoC MPFS###T devices.
2. Once the PFSOC_SCSM macro is instantiated using Libero SoC v2025.1 or later, the SC_WAKE input cannot be tied to a static "1" or "0" logic value because it is an invalid use-model and will trigger a design Compile error in Libero SoC.
3. It is the user's responsibility to perform handshaking between MSS application code and fabric logic if both methods are used to request system services in the same design.

The following figure is an example showing the SC_WAKE input connected to user signals in an RT PolarFire SoC SmartDesign canvas. Such a connection would generate an error for a terrestrial PolarFire SoC design since the MPFS devices only allow the connection from REBOOT_REQUESTED_M2F to SC_WAKE.

Figure 3-1. Connecting SC_WAKE Input to Dynamic User Signal in RT PolarFire SoC Design

For examples of how to use system services, see [PolarFire Family System Services User Guide](#).

3.2. In-Flight Reprogramming [\(Ask a Question\)](#)

For RTPF500T devices, only JTAG and SPI target re-programming modes are supported for in-flight reprogramming, and the process generally involves the use of an external device that executes [DirectC](#). Power-cycle initiated auto-updates are not recommended for RTPF500T devices. Furthermore, system services for In-Application Programming (IAP) and auto-update are not supported because the device must be configured to use SCSM in flight.

The enhancements to RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices add robustness checks to the IAP and auto-update programming algorithms, allowing them to be used for in-flight reprogramming. These checks include the addition of an extra programming verification step and a programming digest check before the FPGA fabric is released for operation. When using IAP or auto-update, it is important to understand the automatic recovery process triggered in the event of a programming failure or interruption, as documented in the [RT PolarFire Programming User Guide](#). RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices also continue to support the JTAG and SPI target re-programming modes.

Notes:

- When reprogramming RTPF500T or RTPF500ZT devices, the sNVM cannot be reprogrammed independently of the FPGA fabric.
- Before considering in-flight reprogramming of RTPFS460ZT or RTPFS160ZT devices using bitstreams that do not contain a fabric component, such as eNVM-only or sNVM-only bitstreams, it is important to review the Recommendations for Field Updates (IAP and Auto Update) table in the [PolarFire and PolarFire SoC Programming User Guide](#).
- The One-Way Passcode (OWP) protocol for PolarFire SoC and RT PolarFire SoC devices is not compatible with the Secure Production Programming Solution (SPPS) that uses Hardware Security Modules (HSMs) and an Authorization Code bitstream component. Programming bitstreams that use OWP are intended for use outside the SPPS environment on devices that have already been provisioned with the desired security settings and locks. The OWP allows for the temporary unlocking of an individual security lock to enable a programming action, such as IAP. Attempting to use a bitstream with OWP in an SPPS environment with Authorization Code bitstream components will result in an error. For more information about OWP, see the [PolarFire and PolarFire SoC Programming User Guide](#). For more information on SPPS and Authorization Code components, see the [SPPS User Guide](#).

For more information about the recommended programming action, such as the REPROGRAM_INFLIGHT action for in-flight reprogramming, see [RT PolarFire Programming User Guide](#) and [DirectC User Guide](#).

The following table summarizes the SCSM exit behavior differences between MPF###T, RTPF500T, RTPF500ZT, MPFS###T and RTPFS###ZT devices, as described in the preceding sections.

Table 3-1. System Controller Suspend Mode (SCSM) Exit Behavior Differences

Device	Exit SCSM through JTAG TRSTB = HIGH for Device Reprogramming	Exit SCSM through JTAG TRSTB = HIGH for System Controller Services	Exit SCSM through FPGA Fabric (PFSOC_SCSM macro) for MSS_REBOOT_REQ	Exit SCSM through Fabric Macro for System Controller Services	INIT_MONITOR Outputs Clear at SCSM Exit
PolarFire® (MPF###T)	Yes (for terrestrial environments)	No	N/A	N/A	Yes
RTPF500T	Yes, for JTAG or SPI Target In-Flight Reprogramming Only	No	N/A	N/A	Yes
RTPF500ZT	Yes	Yes	N/A	N/A	No
PolarFire SoC (MPFS###T)	Yes (for terrestrial environments)	No	Yes	No	Yes
RT PolarFire SoC (RTPFS160ZT, RTPFS460ZT)	Yes	Yes	Yes	Yes	No

4. Radiation Tolerance [\(Ask a Question\)](#)


The following table summarizes the radiation tolerance differences between RTPPF500T and RTPPF500ZT.

Table 4-1. Radiation Tolerance Differences

Device Characteristic	RTPPF500T	RTPPF500ZT
Total Ionizing Dose (TID) performance	Maintains datasheet parameters at 100 krad	
FPGA Configuration Upsets (Firm Errors), including μ PROM	Zero observed at LET 80 MeV-cm ² /mg	
Flip-Flop SEU rate without synthesized TMR	1.3 × 10 ⁻⁷ errors/bit-day (GEO solar min)	
Flip-Flop SEU rate with synthesized spatial TMR	1.2 × 10 ⁻¹¹ errors/bit-day (GEO solar min)	
Fabric LSRAM block SEU rate without ECC	4.7 × 10 ⁻⁸ errors/bit-day (GEO solar min)	
Fabric LSRAM block SEU rate with optional SECDED ECC enabled	4.7 × 10 ⁻¹⁵ errors/bit-day (GEO solar min)	
Reset circuit SEFI rate	Estimated at 1 spontaneous, self-recoverable, device reset in 187 years (GEO solar min)	
Fabric μ SRAM block SEU rate without EDAC	2.3 × 10 ⁻⁸ errors/bit-day (GEO solar min)	
Fabric μ SRAM block SEU rate using CoreEDAC soft IP	9.5 × 10 ⁻¹⁰ errors/bit-day (GEO solar min)	
SEU rate for internal NVM storage including user sNVM and system controller pNVM For more information, see the "pNVM/ boot up failure Results" section in the RT PolarFire RTPPF500ZT Heavy Ion Test Results radiation report.	Exhibits 1 failure in 362,397 years of operation (GEO solar min)—sNVM upset impacts sNVM data integrity and pNVM upset impacts device power-up to functional process	Zero observed NVM data errors at LET = 80 MeV-cm ² /mg
SEL for HSIO bank	No SEL observed on HSIO	
SEL for GPIO bank at 3.3V ± 5% VDDI and VDDAUX	SEL LET _{TH} > 25 MeV-cm ² /mg at room temp. Destructive SEL at LET = 48 MeV-cm ² /mg at 87°C	SEL LET _{TH} = 37 MeV-cm ² /mg. Destructive SEL at LET = 45 MeV-cm ² /mg at 100°C
SEL for GPIO bank at 2.5V ± 5% VDDI and VDDAUX	SEL LET _{TH} = 58 MeV-cm ² /mg	SEL LET _{TH} = 80 MeV-cm ² /mg
SEL for GPIO bank at 1.8V ± 5% VDDI and 2.5V ± 5% VDDAUX	SEL LET _{TH} ≥ 58 MeV-cm ² /mg	SEL LET _{TH} = 80 MeV-cm ² /mg
SEL for XCVR bank REFCLK inputs at 2.5V ± 5% VDD_XCVR_CLK	SEL LET _{TH} = 58 MeV-cm ² /mg	SEL LET _{TH} = 80 MeV-cm ² /mg
micro-SEL for fabric ESD clamp	Fabric ESD clamps exhibit micro-SEL on VDD25/VDDA25 (12 to 30 mA per micro-SEL) with 1 micro-SEL every 27 years in GEO solar min	No fabric ESD clamp micro-SEL observed on VDD25/VDDA25 at LET = 80 MeV-cm ² /mg

In short, the RTPPF500ZT contains enhancements to the silicon design that achieve the following improvements over the initial RTPPF500T devices:

- Elimination of SEL for GPIO bank I/Os using 2.5V ± 5% VDDI and VDDAUX at LET = 80 MeV-cm²/mg
- Elimination of fabric ESD clamp micro-SEL on VDD25/VDDA25 at LET = 80 MeV-cm²/mg
- Zero observed NVM data errors at LET = 80 MeV-cm²/mg
- Confirmed by heavy-ion beam testing. For more information on radiation tolerance of other device features, see the latest test reports on the [Radiation Test Report](#) web page.

 **Important:** The guidelines for high-frequency decoupling capacitor usage, for I/O bank power supplies on the RTPF500T, must be strictly followed, as stated in the [RT PolarFire Board Design User Guide](#).

The RT PolarFire SoC RTPFS460ZT and RTPFS160ZT devices include the same radiation tolerance enhancements as the RTPF500ZT. Therefore, the enhanced SEL performance measured on the RTPF500ZT, as shown in the preceding table, are expected to apply to RTPFS460ZT and RTPFS160ZT devices as well. Once confirmed through radiation testing, the table above will be updated further to include the RT PolarFire SoC devices. Similarly, RTPFS460ZT and RTPFS160ZT devices have the same FPGA fabric radiation performance as the RTPF500ZT.

As mentioned in [Table 1-2](#), the same MSS design is used for both PolarFire SoC and RT PolarFire SoC devices. However, with the addition of ECC on the internal sNVM, RTPFS460ZT, and RTPFS160ZT devices support an enhanced, radiation-mitigated, secure boot mode 2, compared to MPFS###T devices. In this boot mode, the following events occur:

1. The initial bootloader is stored in ECC-protected sNVM.
2. The user boot code is encoded with software-defined ECC before being stored in the device eNVM.
3. During boot-up, the system controller copies the sNVM bootloader into E51 instruction memory, and the E51 monitor core executes the initial bootloader.
4. The initial sNVM bootloader decodes and copies the user eNVM boot code into ECC protected L2 scratchpad memory, correcting single bit errors along the way.
 - Double-bit errors can flag an error to an external host or trigger an IAP from a golden image.
5. The MSS then executes the user eNVM boot code from ECC protected L2 scratchpad memory.

An application note and reference design will be published to release this enhanced boot-mode 2 for RT PolarFire SoC devices.

For the results of the MSS Radiation Tests, see the latest reports on the [Radiation Test Report](#) web page.

5. Device Compatibility Between RTPF500T and RTPF500ZT [\(Ask a Question\)](#)

This section discusses the compatibility of designs targeting RTPF500T and RTPF500ZT. The enhancements made to the RTPF500ZT did not change the design of the FPGA fabric or the CG1509 package.

5.1. Package Compatibility [\(Ask a Question\)](#)

RTPF500T and RTPF500ZT devices in the 1509-pin Ceramic Column Grid Array (CCGA) package are pin and footprint compatible. The CG1509 package design is identical between the two devices. Therefore, a board designed for the RTPF500T in CG1509 package can also use the corresponding RTPF500ZT in CG1509 package without requiring any board changes. Furthermore, the user's top-level design pin assignments for RTPF500T do not require any changes when migrating to RTPF500ZT.

The RTPF500ZT device is also available in a high-reliability plastic FC(G)1509 package. The RTPF500ZT in FC(G)1509 is pin compatible with both the RTPF500T and RTPF500ZT devices in the CG1509 package. For screening flow details and differences between the FC(G)1509 versus the CG1509, see [RT PolarFire FPGA Product Overview](#).

5.2. Bitstream Compatibility [\(Ask a Question\)](#)

RTPF500T and RTPF500ZT programming bitstreams are not compatible with each other. When designing for RT PolarFire FPGAs in Libero SoC, the user must select the appropriate device part number in the project settings or new project wizard. Attempting to program an RTPF500ZT device with a programming bitstream designed for RTPF500T, or vice versa, results in an error and the device is not programmed. The addition of SECDED EDAC on RTPF500ZT internal sNVM and pNVM storage necessitated a change to the programming bitstream, preventing bitstream compatibility between the two device versions. The Libero SoC design flow step that generates the design initialization data has been updated to employ SECDED EDAC when the RTPF500ZT is selected.



Important: An RTPF500T design that has used 100% of the internal sNVM storage cannot be directly migrated to RTPF500ZT because of the extra storage overhead required to store user SECDED ECC protected data in sNVM. In this scenario, Libero SoC generates an error during the **Generate Design Initialization Data** design flow step. The user must re-run the **Configure Design Initialization Data and Memories** step to reduce the amount of sNVM data storage used by the design.

5.3. Design Migration from RTPF500T to RTPF500ZT [\(Ask a Question\)](#)

Since the two devices in CG1509 are package pin and footprint compatible, requiring no board changes when migrating from RTPF500T to the corresponding RTPF500ZT, the instructions in [Appendix 1: Device Migration Process from RTPF500T to RTPF500ZT Using Libero SoC](#) can be followed to migrate a completed, placed and routed RTPF500T design to a RTPF500ZT design with no changes in placement or routing.



Important: RTPF500T is not recommended for new designs. RTPF500T is being replaced by RTPF500ZT FPGAs.

6. Device Compatibility Between MPFS###T and RTPFS###ZT Devices [\(Ask a](#)

[Question\)](#)

The PolarFire SoC MPFS devices and RT PolarFire SoC RTPFS###ZT devices share the same FPGA Fabric and MSS design. Therefore, user application code designed to run on the MSS is portable between these devices. Similarly, FPGA fabric user logic is portable between these devices, with similar area and timing results, except for the typical variations possible between tool flow runs. However, as described in this document, several enhancements have been added to RT PolarFire SoC devices to make them more suitable for space applications. In addition to the built-in enhancements, users must also consider what changes are required to software, firmware, and fabric logic designs to incorporate mitigations for upsets, depending on the mission reliability requirements and the device radiation test reports. Such changes could include synthesized local TMR, distributed TMR, and high-reliability FPGA fabric coding styles.

Similarly, the MSS boot mode selected by the user for RTPFS designs could differ from that used by an MPFS design, such as the enhanced boot mode 2 described in the preceding section. Furthermore, when prototyping RTPFS designs using MPFS, or migrating from MPFS to RTPFS, users should also account for the slightly reduced sNVM storage due to built-in ECC, as well as reduced eNVM storage when using software ECC and the enhanced boot mode 2.

6.1. Package Compatibility [\(Ask a Question\)](#)

The following table summarizes the package pin compatibility between RT PolarFire SoC devices and other PolarFire devices.

RT PolarFire SoC Device	Package Pin Compatible With			
	PolarFire SoC (MPFS)	PolarFire FPGA (MPF)	RT PolarFire FPGA (RTPF500T and RTPF500ZT)	RT PolarFire SoC (RTPFS)
RTPFS160ZT in FCV(G)784	Yes; MPFS160T in FCV(G)784I	None	None	None
RTPFS460ZT in FC(G)1509	None	None	None	Yes; RTPFS460ZT in CG1509
RTPFS460ZT in CG1509	None	None	None	Yes; RTPFS460ZT in FC(G)1509

6.2. BitStream Compatibility [\(Ask a Question\)](#)

RT PolarFire SoC (RTPFS) bitstreams are not compatible with PolarFire SoC bitstreams (MPFS). When designing for RT PolarFire SoCs in Libero SoC, the user must select the appropriate device part number in the project settings or new project wizard.

7. I/O Bank Voltages [\(Ask a Question\)](#)

You must be familiar with the Single-Event Latch-up (SEL) risks for GPIO banks as described in [Radiation Tolerance](#) and the [RT PolarFire Radiation Test Reports](#) before selecting 2.5V or 3.3V I/Os.

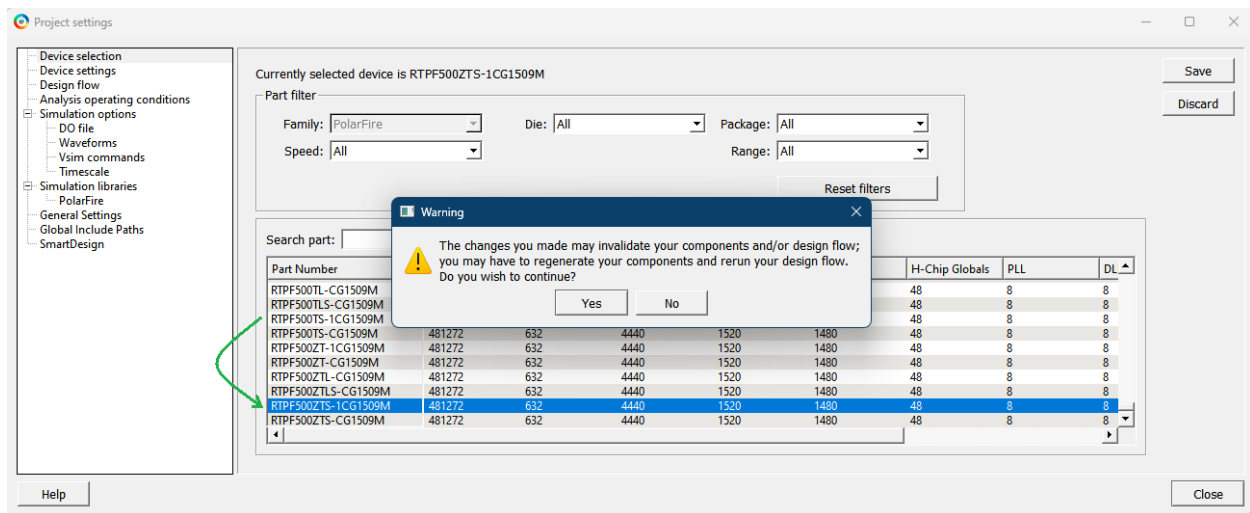
When designing for RTPF500T, RTPF500ZT, RTPFS460ZT, and RTPFS160ZT devices in Libero SoC, the default project settings are configured to generate an error and stop the design flow if 3.3V I/O standards are used on GPIO banks. This setting is accessed from **Project Menu > Project Settings > Design Flow Settings**.

8. Appendix 1: Device Migration Process from RTPF500T to RTPF500ZT Using Libero SoC [\(Ask a Question\)](#)

The enhancements made to the RTPF500ZT do not impact the FPGA fabric design, and thus user design content remains unchanged between the two devices. Therefore, a Libero® SoC project that has been placed and routed for RTPF500T can be updated to target RTPF500ZT when migrating between device versions.

The following figure shows the Libero SoC project settings window, where the users can change the target device die setting for a completed and finalized RTPF500T design.

Figure 8-1. Change Device Selection



When using the Libero SoC managed design flow starting with HDL source files, changing the target device die invalidates the Libero SoC design state back to the pre-synthesis stage. In contrast, when using the [Custom Flow](#), where the final top-level Libero SoC project starts with a synthesized mapped-Verilog netlist (.vnm file), changing the target device invalidates the design to the pre-compile state. When the synthesized netlist is the design source file for the RTPF500T Libero SoC project, the same netlist can be used as the starting point for the updated project targeting the RTPF500ZT, without resynthesizing.

In general, when using the same input source files, computer platform, tool versions, and tool settings (other than the target device), the tool flow can be re-run using the **Incremental Place and Route** setting to recreate the previously completed place and route result.

➔ Important: The following considerations help to ensure a smooth Libero SoC design migration from RTPF500T to RTPF500ZT:

- The final state of the RTPF500T project determines whether the exact place and route results can be repeated after changing the target device selection to RTPF500ZT.
- Ensure that the pin assignment in the final RTPF500T design is locked using **I/O Editor** or a PDC file with `-fixed true` argument on each `set_io` constraint.
- Ensure that all logical instances in the final RTPF500T design have a locked placement in the **Chip Planner** tool and that the corresponding user floor-planning PDC file is enabled in the Constraint Manager.

- In general, it is recommended to archive (zip) the original completed RTPF500T project before modifying the target device.
- Do not change the Libero SoC or Synopsys® Synplify Pro® tool versions when migrating the design.
- Do not change project or tool settings other than the device die selected during the migration.
- Do not change any input source files, constraint files, or file organization/associations during the migration.
- For consistent and repeatable results, ensure the migrated RTPF500ZT project design flow is run using the same OS and comparable CPU type (for more information, see [Note](#)) as the computer used to create the final RTPF500T design.
- Plan ahead for migration from RTPF500T to RTPF500ZT by starting the final top-level RTPF500T Libero SoC project using the synthesized .VM netlist along with the final constraint files. This avoids re-synthesizing the HDL source files during the design migration, ensuring the input source files to the Libero SoC incremental place and route process are unchanged.

The following figures from the **I/O Editor** accessed by selecting **Edit** in the **Constraints Manager's I/O Attributes** tab, highlight all top-level I/O pins and I/O bank settings in the RTPF500T design are locked:

Figure 8-2. I/O Editor Pinout Locked

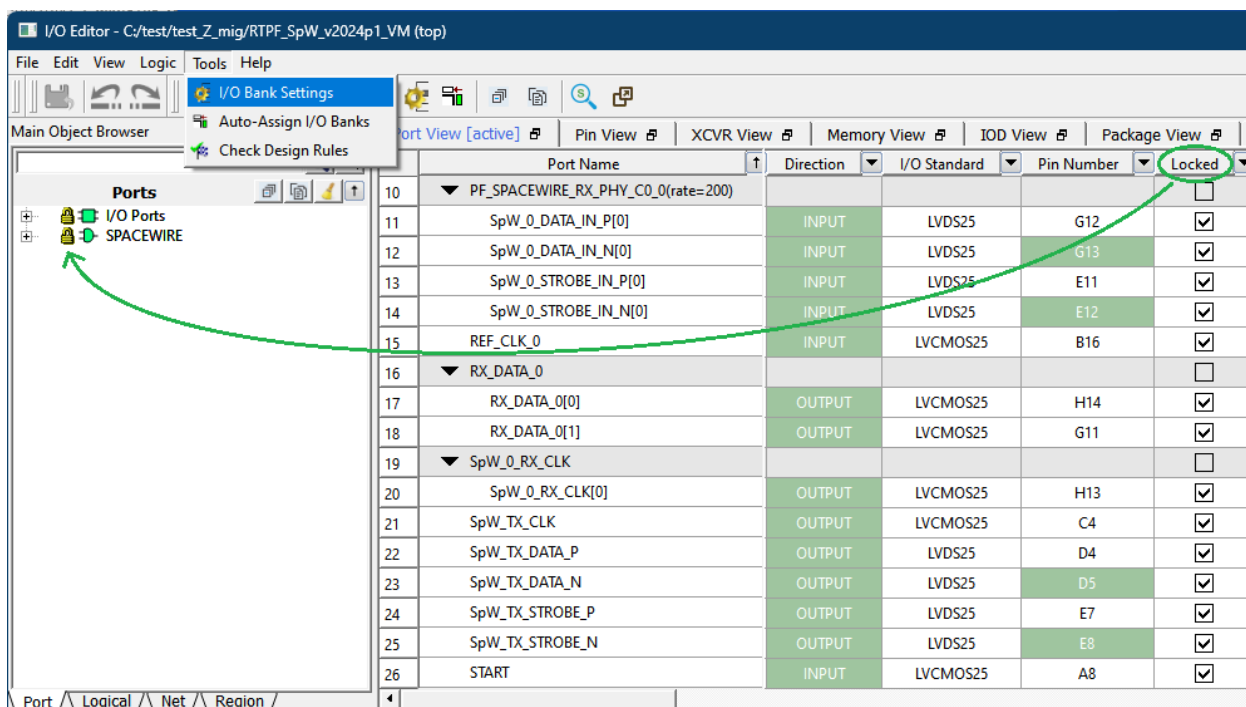


Figure 8-3. I/O Bank Settings Locked



The corresponding I/O PDC file used in the Constraint Manager contains set_io constraints with argument `-fixed true`.

Figure 8-4. User PDC with set_io -fixed true

```

99
100 set_io -port_name {SpW_0_STROBE_IN_N[0]} \
101 -pin_name E12 \
102 -fixed true \
103 -io_std LVDS25 \
104 -DIRECTION INPUT \
105
106
107 set_io -port_name {SpW_0_STROBE_IN_P[0]} \
108 -pin_name E11 \
109 -fixed true \
110 -io_std LVDS25 \
111 -DIRECTION INPUT \

```

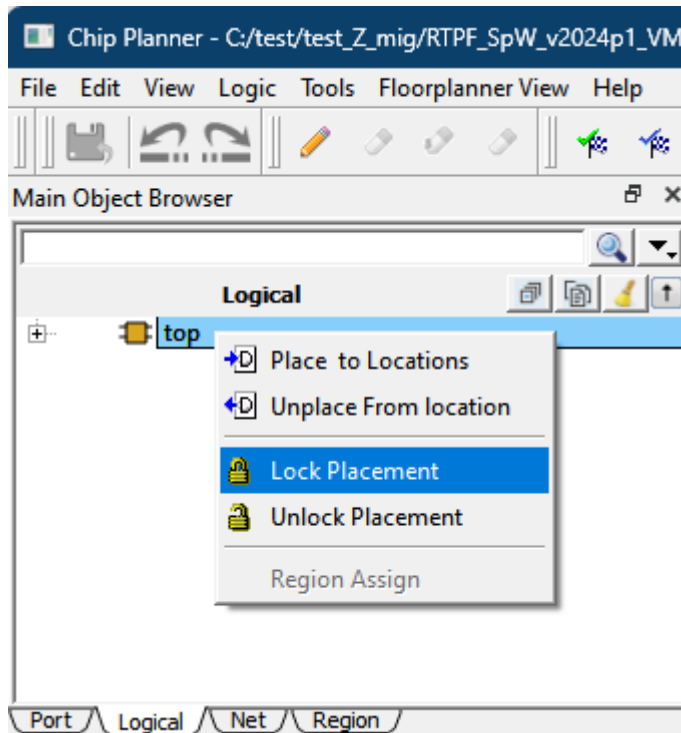
The design pin report also reflects that the user pin assignment is fixed.

Figure 8-5. Pin Report Fixed I/O

Port	Pin	Fixed	Function
Clk_err_Inj	A5	Yes	GPIO307PB2/
cnten	C3	Yes	GPIO309PB2/
cntout[24]	E14	Yes	GPIO1NB2
cntout[25]	E13	Yes	GPIO1PB2
cntout[26]	F14	Yes	GPIO0NB2
cntout[27]	F13	Yes	GPIO0PB2

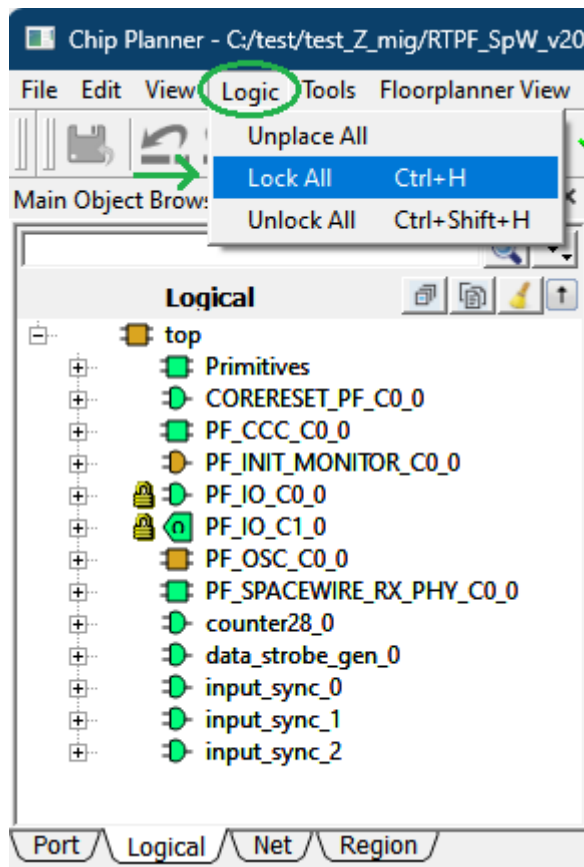
The following figures from the **Chip Planner** accessed by selecting **Edit** in the **Constraint Manager** from **Floor Planner** tab, highlight how the user can ensure the final RTPF500T design locks the placement of all logical instances and that the floor-planning `user.pdc` file written out by the Chip Planner is enabled for use in the design.

The user can right-click the hierarchical root node in the Logical view of the Main Object Browser in Chip Planner to Lock Placement or use the **Logic** Menu -> **Lock All** option from the menu bar.

Figure 8-6. Lock Placement through Chip Planner Right-Click on Root Logical Node

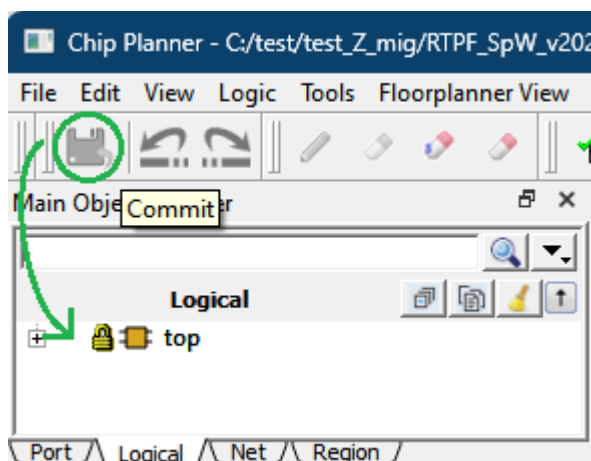
OR

Figure 8-7. Using Chip Planner Logic Menu to Lock All Instance Placement



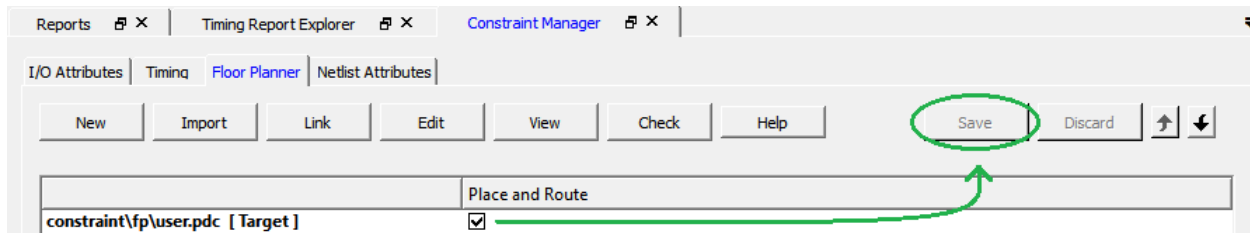
Use the **File > Commit** or click the toolbar icon to save the change and write-out a floor-planning `user.pdc` file:

Figure 8-8. Commit Placement Chip Planner



The `user.pdc` file is automatically saved into `<path_to_project>/constraint/fp/user.pdc`

Enable the usage of the floor-planning user PDC containing the locked placement for all logical instances in the **Constraint Manager > Floor Planner** tab:

Figure 8-9. Enable Usage of Fixed Placement PDC and Save Constraint Manager Floor Planner

The following figure shows an example of the floor-planning PDC file generated by locking the placement of all logical instances in the completed design.

Figure 8-10. Placement PDC Fixed True

```
# Microchip Physical design constraints file
# Version: 2024.1 2024.1.0.3
# Design Name: top
# Family: PolarFire , Die: RTPF500TS , Package: CG1509 , Speed grade: -1
#
# Core cell constraints
#
set_location -inst_name PF_INIT_MONITOR_CO_0/PF_INIT_MONITOR_CO_0/I_Suspend_Mode_Soft_IP_Inst/1
true -x 414 -y 4
set_location -inst_name PF_CCC_CO_0/PF_CCC_CO_0/clkint_0/U0_RGB1 -fixed true -x 579 -y 12
set_location -inst_name {data_strobe_gen_0/strbo_err_int_0[1]} -fixed true -x 146 -y 3
set_location -inst_name {CORERESET_PF_CO_0/CORERESET_PF_CO_0/dff_5[0]} -fixed true -x 177 -y 4
```

Once the preceding considerations have been made, rerunning the incremental Libero SoC design flow to generate the final RTPF500T design bitstream puts the design into a state that ensures the user can obtain the same place and route results after migrating the project's device die setting to the corresponding RTPF500ZT device.

The following is a summary of the migration steps.

1. Prerequisites for Migration and RTPF500T Final Design State:

- Use the same Libero SoC and Synplify Pro tool version.
- Use the same input source files, constraint files, file organization, and associations.
- Use the same tool settings, other than the device die setting which is being migrated.
- Run the design software on the same OS and CPU type (for more information, see [Note](#)).
- The final pin assignment must be locked in the graphical I/O Editor or have equivalent set_io commands with `-fixed true` and the I/O `user.pdc` file must be enabled in the Constraint Manager.
- The finalized RTPF500T design bitstream must be obtained after rerunning the incremental design flow with locked instance placement and the corresponding user floor-planning PDC enabled in the Constraint Manager. That final project state must be achieved before using the RTPF500T design to perform final hardware/system testing and before migrating to RTPF500ZT to ensure the same fabric Place and Route result and fabric digest after migration.



Tip: In general, different CPU generations should produce the same results for floating-point arithmetic used in FPGA Place and Route algorithms, if those CPUs use the same precision and follow the IEEE®-754 standard for floating point arithmetic. However, when running an FPGA Place and Route algorithm on different CPU architectures, it is theoretically possible to arrive at different place and route results due to small computational differences. For the specific RTPPF500T to RTPPF500ZT migration guidance in this document, where the user is instructed to use a locked placement and incremental Place and Route, the impact of the CPU type should be minimized.

2. **Libero SoC Design Migration Steps:** To migrate the Libero SoC design, perform the following steps:
 - a. In Libero SoC, open the final RTPPF500T design (ensuring that a backup copy exists)
 - b. To change the Device Selection to the corresponding RTPPF500ZT die, navigate to the **Project > Project Settings...** window and save the changes.
 - The design state will be invalidated to pre-Synthesis state (if the design started with HDL source files) or pre-Compile state (if the design started with a synthesized netlist (.VM)).
 - c. To enable Incremental Place and Route, right-click the design flow step and select **Configure Options...**
 - d. For the Libero SoC managed design flow, where the project uses RTL source files as inputs, re-run the Libero SoC design flow, including re-running Synthesis, Incremental Place and Route, and Generate Bitstream.
OR
 - e. For the **Custom Flow**, where Synthesis is performed outside of Libero SoC and the synthesized .VM netlist is the input source file to the Libero project, do not re-run Synthesis. Instead, re-run the Libero SoC design flow from netlist Compile, including Incremental Place and Route, and Generate Bitstream design flow steps.
 - f. Review the following files to confirm that the place and route result is unchanged between the RTPPF500T and RTPPF500ZT designs:
 - i. The layout log for the RTPPF500ZT project must indicate that there were 0 nets incrementally routed. The log is found at: `<path_to_project>/Designer/<top_name>/<top_name>_layout_log.log`
 - ii. The `<top_name>_generateBitstream.log` contains the design CHECKSUM and Fabric component bitstream digest. Compare these values in the migrated RTPPF500ZT project to the RTPPF500T project. If these values are the same, it means the two projects have used the same place and route results and the fabric content in the programming bitstream is identical.
 - g. Complete the appropriate design flow steps under **Hand-off Design for Production** to ensure that the files used for production hand-off are specific to the RTPPF500ZT device. This includes re-exporting the **Design Initialization Data and Memory Report** to obtain updated design-specific power-up to functional time (PUFT) estimates. After migrating to the RTPPF500ZT, re-exporting this report will account for the added PUFT overhead of decoding the SECDED ECC protected sNVM initialization data during device initialization performed by the internal system controller. As long as the design follows published guidance in the [PolarFire Family Power-Up and Resets User Guide](#) and [PolarFire FPGAs for Safety-Critical Applications](#) application note regarding device initialization process, PolarFire Initialization Monitor output signal usage, and latching system controller outputs during suspend mode, the user design will be held in reset until the device initialization completes.

The following figures show how to enable Incremental Place and Route.

Figure 8-11. Configure Layout Options

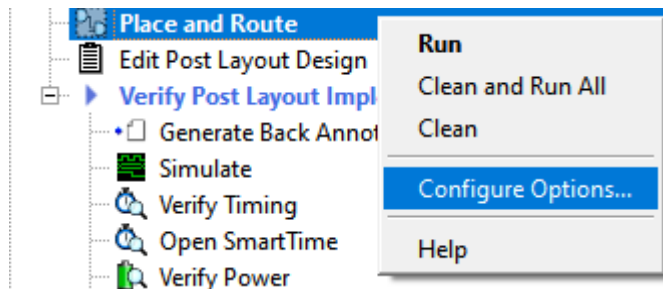
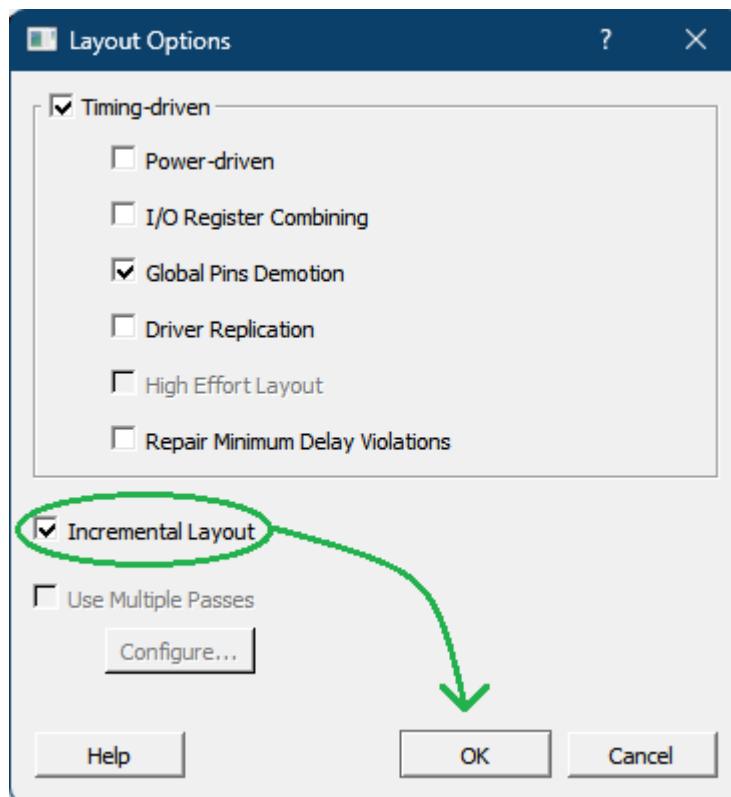


Figure 8-12. Enable Incremental Place and Route



The following figures show excerpts from the layout log during Incremental Place and Route where 0 nets are re-routed.

Figure 8-13. Layout Log 1

```
***** Place and Route Configurations *****
Timing-driven           : ON
Power-driven            : OFF
I/O Register Combining  : OFF
Global Pins Demotion    : ON
Driver Replication      : OFF
High-effort             : OFF
Repair min-delay        : OFF
Incremental             : ON ←
Inter-clock optimization : ON
```

Figure 8-14. Layout Log 2

```
Running the I/O Bank and Globals Assigner.
Info: I/O Bank and Globals Assigner is running in incremental mode. Pre-assigned I/O Bank technologies and cells will not be changed
Info: I/O Bank and Globals Assigner identified 18 fixed I/O macros, 0 unfixed I/O macros
Info: I/O Bank and Globals Assigner identified bank 'Bank2' as being fixed at VCCI:2.50V VCCR:n/a
```

Since the RTPF500T final design includes a fixed placement floor-planning PDC for all logical instances, and the input source files have not changed in the RTPF500ZT project, the placer completes quickly, as shown in the following log excerpt.

Figure 8-15. Layout Log 3

```
Placer Runtime Summary      :
Clustering (1 pass)         : 0 seconds
Placement                   : 0 seconds
Improvement                 : 0 seconds

Placer completed successfully.
```

Figure 8-16. Layout Log 4

```
Info: Routed 0 net(s) incrementally.

Router completed successfully.
```

The following figure shows an example of the generateBitstream.log highlighting the design CHECKSUM and Fabric component bitstream digest values that must be compared for consistency across the RTPF500T and migrated RTPF500ZT designs.

Figure 8-17. Generate Bitstream Log

```
Software Version: 2024.1.0.3
Opened 'C:\test\test_Z_mig\RTPF_SpW_v2024pl_VM\designer\top\top_fp\top.pro'
PDB file 'C:\test\test_Z_mig\RTPF_SpW_v2024pl_VM\designer\top\top.pdb' has been loaded successfully.
DESIGN : top; CHECKSUM : D893; PDB_VERSION : 1.9
Load Programming Data(s) Finished : Thu Jun 06 12:14:44 2024 (Elapsed time 00:00:00)
Info: Programming Interface selected is JTAG.
File/Folder 'C:\test\test_Z_mig\RTPF_SpW_v2024pl_VM\designer\top\top.ppd' will be overwritten.
Successfully exported PPD file for currently secured device: 'C:\test\test_Z_mig\RTPF_SpW_v2024pl_VM\designer\top\top.ppd';
file programs Fabric and sNVM.
BITS component bitstream digest: f185dcdc588fcd25cf2e26aa6b39b317c7cd244655399a122a409fb252d8e23
Fabric component bitstream digest: fd11cb8913ff18c891d9eb521a73e0dd2d6342e24e30cd742a5952de43af0388
sNVM component bitstream digest: 9017595a7babaf612b12e90cb5d96343fc65f6e453f7517ad990fa33cfell1445
EOB component bitstream digest: 2a052ab24ddd57012177919b1d0ed8d4c49ea037b865914dbf6a067329240a74
Entire bitstream digest: 96f6b7beb46f2115c76f3606a6ce5a4f8097c59462bd9f451371b054a4bf7588
Finished: Thu Jun 06 12:15:49 2024 (Elapsed time 00:01:05)
```

9. Conclusion [\(Ask a Question\)](#)

This application note is intended to clearly distinguish the supported features and radiation performance of the RT PolarFire RTPF500T and RTPF500ZT devices. Similarly, this document highlights the enhancements made to the RT PolarFire SoC RTPFS460ZT and RTPFS160ZT devices over MPFS###T devices. Planning the system design according to the documented device specific capabilities allows you to choose the device most suited for the application requirements, operational environment, and production schedule.

10. Revision History [\(Ask a Question\)](#)

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

Revision	Date	Description
H	01/2026	<p>The following is the list of changes made in this revision.</p> <ul style="list-style-type: none"> Added a note about RT PolarFire® RTPF500ZT Heavy Ion Test Result and cross referenced the same in Table 1-1 and Table 1-2 in the High-Level Differences section. Made the following changes in the Radiation Tolerance section: <ul style="list-style-type: none"> Updated Table 4-1 with RT PolarFire® RTPF500ZT Heavy Ion Test Result weblink. Updated the note by clearly stating that the decoupling capacitors usage for I/O Bank power supplies must be strictly followed on RTPF500T.
G	10/2025	<p>The following is the list of changes made in this revision.</p> <ul style="list-style-type: none"> From Revision G onwards, the Errata section describing the usage of external VERIFY_DIGEST JTAG/SPI action for fabric component digest on devices configured to perform POR fabric digest checks has been moved to the RT PolarFire FPGA and SoC FPGA Devices Errata document. Added a note regarding RTPF500T device in the Introduction and Design Migration from RTPF500T to RTPF500ZT sections. Figure 3-1 was updated to use an OR gate to combine multiple sources that drive the SC_WAKE input to the PFSOC_SCSM macro.
F	07/2025	<p>The following is the list of changes made in this revision.</p> <ul style="list-style-type: none"> The document was updated to include enhancements in RT PolarFire® SoC RTPFS460ZT and RTPFS160ZT devices compared to other PolarFire devices. The document title was changed from "Differences Between RT PolarFire RTPF500T and RTPF500ZT FPGAs" to "RT PolarFire Device Family Enhancements Compared to PolarFire Devices". Added section for describing a limitation on the usage of external VERIFY_DIGEST JTAG/SPI action for fabric component digest on devices configured to perform fabric Power-On-Reset (POR) digest checks and a section for Processing MSS Reboot Requests During SCSM. Table 4-1 was updated to include the temperature test conditions for the SEL data on GPIO banks running at 3.3V in the Radiation Tolerance section. The steps to migrate a completed design from RTPF500T to RTPF500ZT were moved into Appendix 1.
E	12/2024	<p>The following is the list of changes made in this revision.</p> <ul style="list-style-type: none"> Added Transceiver VDDA supply voltage difference in Table 1-1. Updated System Controller Suspend Mode (SCSM) to indicate that System Controller Suspend Mode (SCSM) can be exited by driving JTAG TRSTB High. "System Controller Suspend Mode (SCSM) Exit Behavior Differences" table was added for summarizing the SCSM exit behavior differences in the In-Flight Reprogramming section. Updated Table 4-1 in the Radiation Tolerance section to reflect the latest SEL LET thresholds for 3.3V GPIO banks on RTPF500T, as per the RT PolarFire FPGA Single Event Latch-Up Test Report Revision B.
D	10/2024	<p>The following is the list of changes made in this revision.</p> <p>Updated Design Migration from RTPF500T to RTPF500ZT section as follows:</p> <ul style="list-style-type: none"> Added a note under the Prerequisites for Migration and RTPF500T Final Design State: in Design Migration from RTPF500T to RTPF500ZT, to clarify the meaning of using the same CPU type as that used to complete the RTPF500T design, when re-running the tool flow for the migrated RTPF500ZT design. Added a final step to Libero SoC Design Migration Steps in Design Migration from RTPF500T to RTPF500ZT section, as a reminder to re-export the migrated RTPF500ZT design specific files for production hand-off, including the Design Initialization Data and Memory Report.

Revision History (continued)		
Revision	Date	Description
C	07/2024	<p>The following is the list of changes made in this revision:</p> <ul style="list-style-type: none"> Added Device Compatibility Between RTPF500T and RTPF500ZT section that includes a new section on Package Compatibility, the pre-existing section on Bitstream Compatibility Updated Libero SoC Design Migration Steps in Design Migration from RTPF500T to RTPF500ZT section.
B	05/2023	<ul style="list-style-type: none"> In Table 1-1, added a row to highlight the difference in sNVM data storage capacity for storing plaintext and ciphertext. Updated Table 4-1, as follows: <ul style="list-style-type: none"> Clarified that the Configuration Upset immunity applies to both the FPGA fabric and the μPROM configuration cells. Added a row describing SEL results for GPIO bank VDDI and VDDAUX at $3.3V \pm 5\%$, in RTPF500T and RTPF500ZT. Added a row describing SEL results for GPIO bank VDDI = $1.8V \pm 5\%$ and VDDAUX = 2.5V, in RTPF500T and RTPF500ZT. In Radiation Tolerance, added a note that points to the guidelines to be followed for high frequency decoupling capacitor usage for I/O bank power supplies. In Radiation Tolerance and I/O Bank Voltages, added link to Radiation Test Reports.
A	02/2023	Initial Revision

Microchip FPGA Support

Microchip FPGA products group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, and worldwide sales offices. Customers are suggested to visit Microchip online resources prior to contacting support as it is very likely that their queries have been already answered.

Contact Technical Support Center through the website at www.microchip.com/support. Mention the FPGA Device Part number, select appropriate case category, and upload design files while creating a technical support case.

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call **800.262.1060**
- From the rest of the world, call **650.318.4460**
- Fax, from anywhere in the world, **650.318.8044**

Microchip Information

Trademarks

The “Microchip” name and logo, the “M” logo, and other names, logos, and brands are registered and unregistered trademarks of Microchip Technology Incorporated or its affiliates and/or subsidiaries in the United States and/or other countries (“Microchip Trademarks”). Information regarding Microchip Trademarks can be found at <https://www.microchip.com/en-us/about/legal-information/microchip-trademarks>.

ISBN: 979-8-3371-2728-6

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP “AS IS”. MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP’S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer’s risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip products:

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip products are strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is “unbreakable”. Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.