# UG0748
# User Guide
# PolarFire FPGA Low Power

Microsemi
Power Matters.

**Microsemi Corporate Headquarters**
One Enterprise, Aliso Viejo,
CA 92656 USA
Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Fax: +1 (949) 215-4996
Email: sales.support@microsemi.com
www.microsemi.com





## About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, California, and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

50200748. 2.0 6/17

# Contents

# Figures

# Tables

# 1 Revision History

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

## 1.1 Revision 2.0

The following is a summary of the changes made in revision 2.0 of this document.

- The document was updated to include features and enhancements introduced in the Libero SoC PolarFire v1.1 SP1 release.
- Information about CCC configuration was updated. For more information, see Configuring CCC, page 4.

## 1.2 Revision 1.0

Revision 1.0 was the first publication of this document.

# 2 PolarFire FPGA Low-Power

PolarFire™ FPGAs offer several low-power modes for the FPGA fabric, transceiver, PCIe, DDR memory, and other hard and soft IP blocks. Flash*Freeze mode is an ultra-low-power static mode with low standby power. The most important low-power mode is Flash*Freeze, which allows easy entry and exit from ultra-low-power static mode while retaining the SRAM content, I/O state, and register data. Flash*Freeze is used in a wide variety of applications requiring low static power to function, such as patient monitoring systems, power distribution systems, industrial, automotive, and mobile applications.

## 2.1 Features

PolarFire devices offer the following low-power features:

- User-configurable Flash*Freeze entry through the CoreSysServices_PF IP.
- User-configurable Flash*Freeze exit through various triggers. For more information, see Flash*Freeze Exit, page 8.
- Retention of state information of the fabric registers, fabric SRAM content, and I/Os leads to fast recovery of the device to active mode.
- Fabric RAMs can be optimized for low power.
- Fabric clock conditioning circuitry (CCC) can be configured to use minimum VCO frequency for low power.

## 2.2 Low-Power Modes

The following low-power modes are available in PolarFire devices:

- Flash*Freeze mode
- Transceiver PMA/PCS low-power mode
- PCIe low-power mode
- DDR low-power mode
- CCC and RAM blocks low-power mode

### 2.2.1 Flash*Freeze Mode

The implementation of Flash*Freeze consists of three phases:

- Flash*Freeze entry
- Flash*Freeze
- Flash*Freeze exit

#### 2.2.1.1 Flash*Freeze Entry

PolarFire devices are designed and optimized to enter Flash*Freeze only when the power supply is stable. Flash*Freeze can be initiated by sending the Flash*Freeze service request command to the system controller through the CoreSysServices_PF IP.

Flash*Freeze entry can be initiated by one of the following methods:

- Using a fabric master logic (Non-AMBA interface)
- Using an APB interface (AMBA interface)

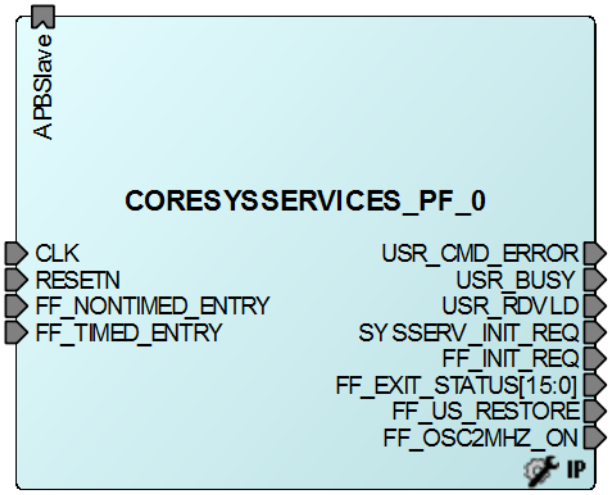
The CoreSysServices_PF IP provides access to the system services supported by PolarFire devices. The system services interface consists of an asynchronous command/response interface, which is used to transfer a system service command from the fabric to the system controller and the status back from the system controller to the fabric.

The system services interface operates in conjunction with the mailbox interface, which is a synchronous interface operating on a user-supplied clock. The mailbox interface is used for passing data related to the system services between the fabric and the system controller. This interface consists of a set of address, data, and control signals for a 2 kB dual-port RAM within the system controller.

The CoreSysServices_PF IP has an arbiter module, which contains priority assessment logic that handles two different Flash*Freeze requests coming from different interfaces (AMBA and non-AMBA) simultaneously.

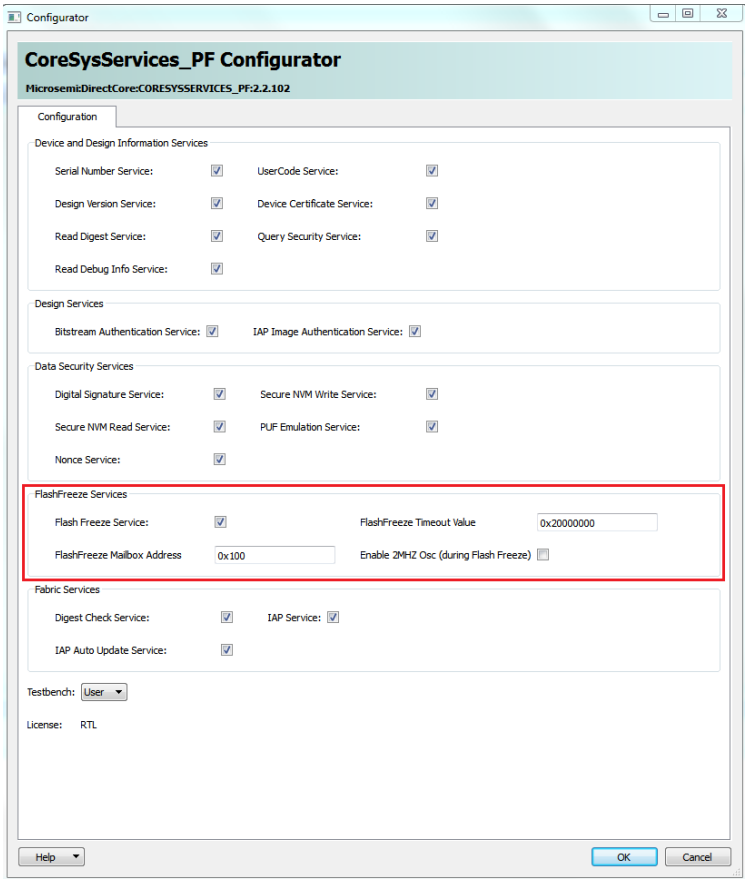
The SmartDesign instantiated view of CoreSysServices_PF IP is shown in the following figure.

***Figure 1 •*** **CoreSysServices_PF IP**




The IP core can be configured using the configurator in SmartDesign, as shown in the following figure.

***Figure 2 •*** **CoreSysServices_PF Configurator**



#### 2.2.1.1.1 Entry Using Fabric Master Logic (Non-AMBA)

The fabric user logic asserts the following signals in the CoreSysServices_PF IP:

- FF_ENTRY_TIMED—used to trigger timed Flash*Freeze entry request.
- FF_NON_TIMED_ENTRY—used to trigger non-timed Flash*Freeze entry request.

The timeout value (~ms) or the amount of time the PolarFire device remains in Flash*Freeze can be specified using the CoreSysServices_PF IP configurator.

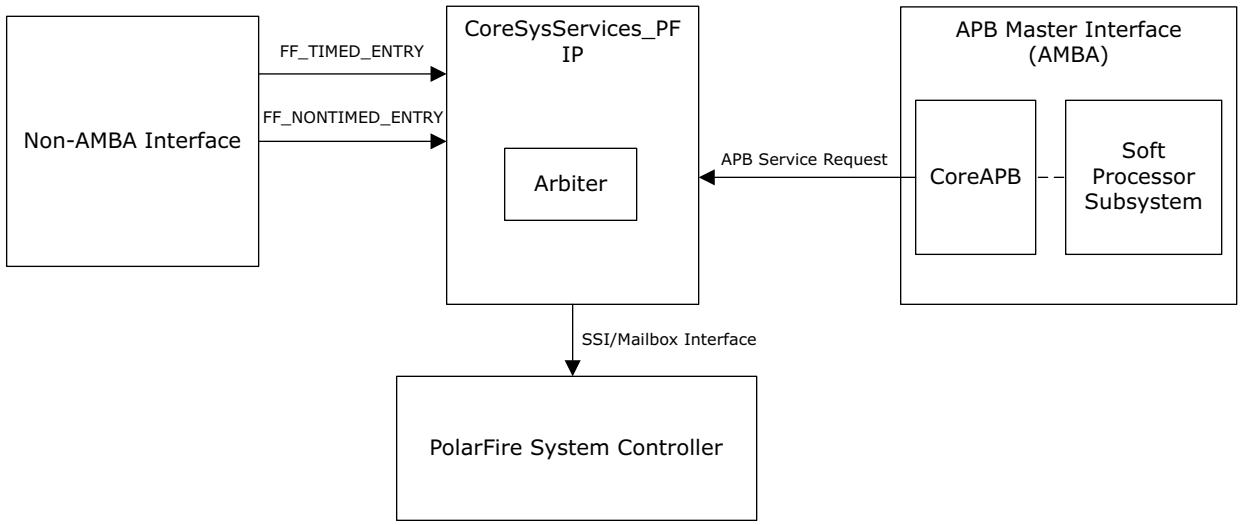
For more information about these signals, see *CoreSysServices_PF IP Handbook (to be released)*.

#### 2.2.1.1.2 Entry Using APB Master Interface (AMBA)

The Flash*Freeze system service request is issued from the APB master by configuring the APB slave registers for timed and non-timed requests.

The following figure shows how AMBA and non-AMBA interface interact with CoreSysServices_PF.

***Figure 3 •*** **AMBA/Non-AMBA Flash*Freeze Requests**

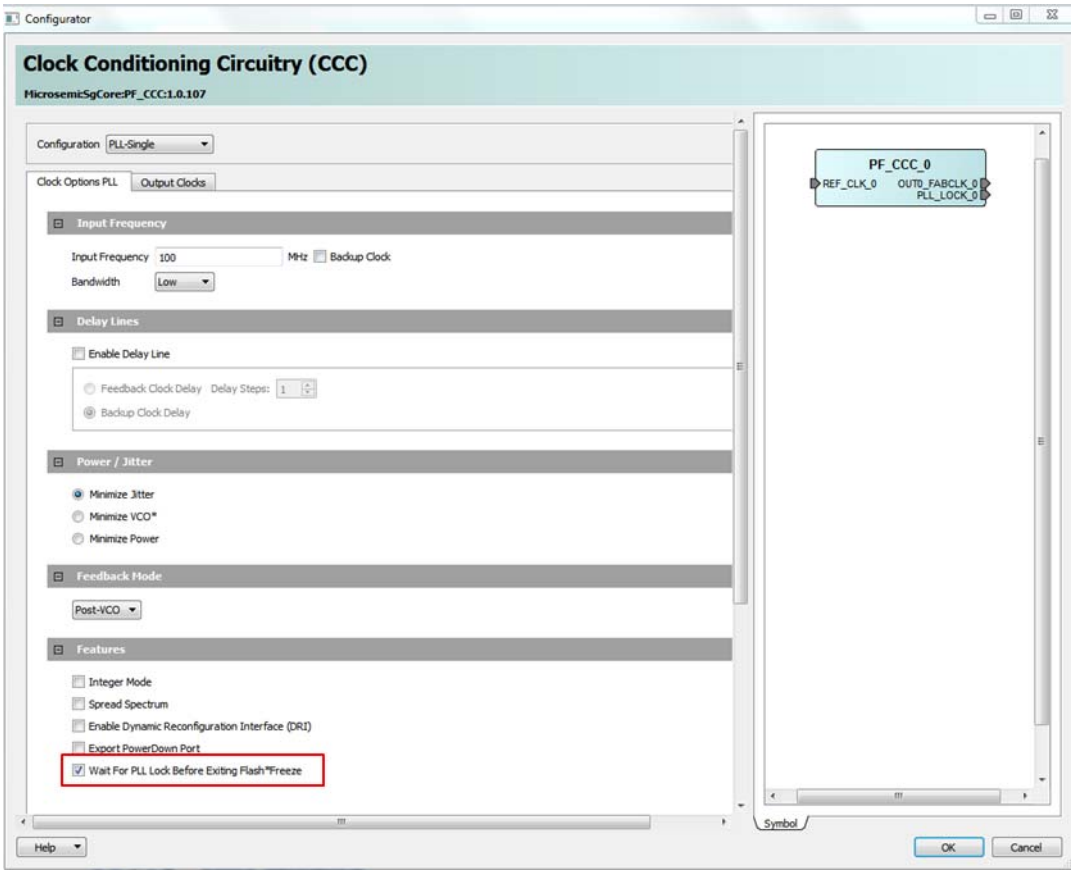



#### 2.2.1.1.3 Configuring CCC

In the PolarFire FPGA CCC Configurator, fabric CCCs can be configured to wait for the PLL lock status before exiting Flash*Freeze, as shown in the following figure. This ensures the system controller waits for the fabric CCCs to be locked before exiting Flash*Freeze and prevents glitches in designs where the PLL lock is used as an asynchronous reset.

**Note:** The CCC, which needs to be locked before Flash*Freeze exit, must always be in internal feedback mode. For more information about internal feedback mode, see *UG0684: PolarFire FPGA Clocking Resources User Guide*.

***Figure 4 •*** **PolarFire FPGA CCC Configurator**

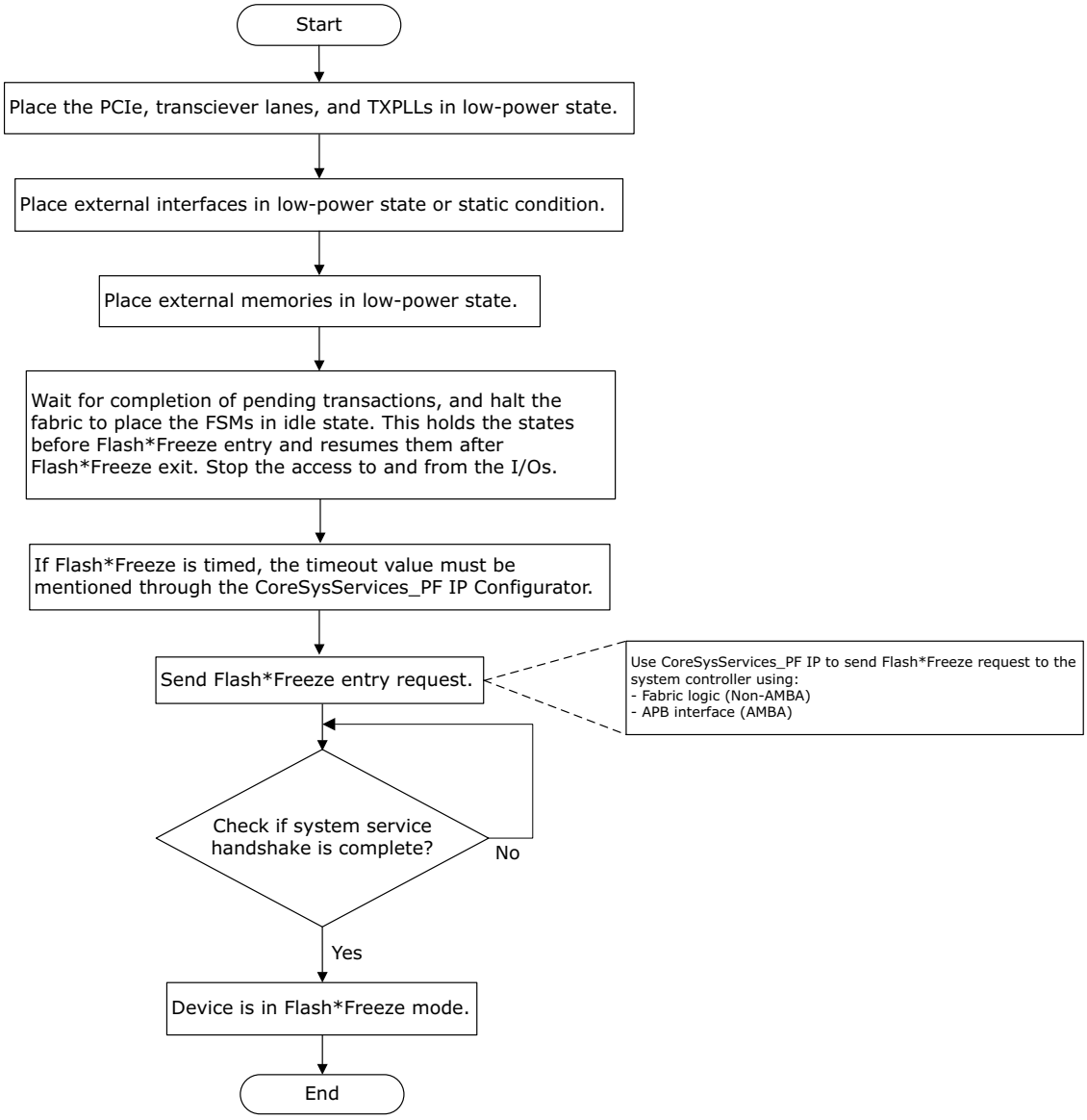



CCC dynamic configuration can be used to divide CCC output clocks on the fly and provide the required power reduction for user applications. For information on how to perform dynamic configuration using dynamic reconfiguration interface (DRI) and related register settings, see *UG0684: PolarFire FPGA Clocking Resources User Guide*.

### 2.2.1.1.4 Entry Sequence

In Flash*Freeze entry, the system controller initiates the sequence of steps shown in the following figure. The fabric master, APB master, or PCIe interface must be interfaced with the CoreSysServices_PF IP to place the device in a low-power state.

***Figure 5 •*** **Flash*Freeze Entry**



### 2.2.1.2 Flash*Freeze

During Flash*Freeze, various blocks of the device are set to different states. I/Os of the device can also be configured to retain their states before Flash*Freeze entry.

#### 2.2.1.2.1 Block States During Flash*Freeze

The following table describes the states of different blocks during Flash*Freeze.

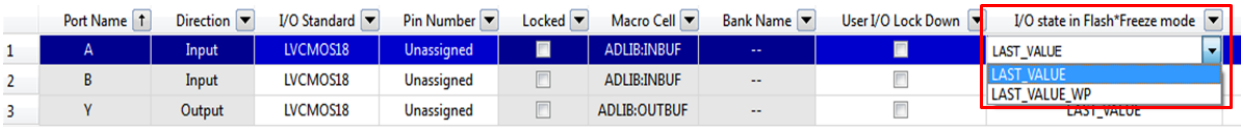
***Table 1 •*** **Block States During Flash*Freeze**

| Block | State |
|---|---|
| System controller | If the 2-MHz oscillator is instantiated in the user design or a timed Flash*Freeze request is sent, the system controller wakes up regularly to scrub the system controller SRAMs. If the 2-MHz oscillator is not instantiated in the user design or a timed Flash*Freeze request is not sent, the system controller is powered down. |
| Secure non-volatile memory (sNVM) | Powered down. |
| Clock glitch detectors | Disabled if clock gating is enabled during Flash*Freeze, otherwise enabled. |
| FPGA fabric | Powered down. |
| FPGA fabric registers | Retain the state. |
| LSRAM | Retain the data. |
| µSRAM | Retain the data. |
| µPROM | Powered down. |
| Mathblocks | Powered down. |
| Transceiver and TXPLLs | User defines the behavior. |
| PCIe | User defines the behavior. |
| GPIO | Tristated, retains last value. |
| HSIO | Tristated, retains last value. |
| Transceiver I/O | User defined based on transceiver behavior. |
| 2-MHz RC oscillator | If the 2-MHz RC oscillator is instantiated in the user design, the oscillator is enabled. Otherwise, it is powered down. |
| 160-MHz RC oscillator | Powered down. |
| Temperature/voltage sensor (TVS) | If the sensor is in voltage detection mode, it is automatically disabled during Flash*Freeze, and if in temperature sensing mode, then it remains enabled during Flash*Freeze. |
| PoR voltage detectors | Powered up. |
| User voltage detectors | Remain in the same state as specified at the design time. Cannot be dynamically disabled or enabled. |
| User glitch detectors | Remain in the same state as specified at the design time. Cannot be dynamically disabled or enabled. |
| CCC | Powered down. |

#### 2.2.1.2.2 I/O State During Flash*Freeze

In the LiberoSoC PolarFire FPGA I/O Editor, each I/O can be configured to hold the previous state, as shown in the following figure.

***Figure 6 •*** **I/O Editor**




The last valid state of the input or output pad can be held before the device enters Flash*Freeze. Weak pull-up of I/O pads can also be configured without affecting the hold state. The following options are available for configuring I/O state during Flash*Freeze:

- LAST_VALUE: holds the previous state of the I/O.
- LAST_VALUE_WP: holds the previous state of the I/O along with weak pull-up without affecting the hold state.

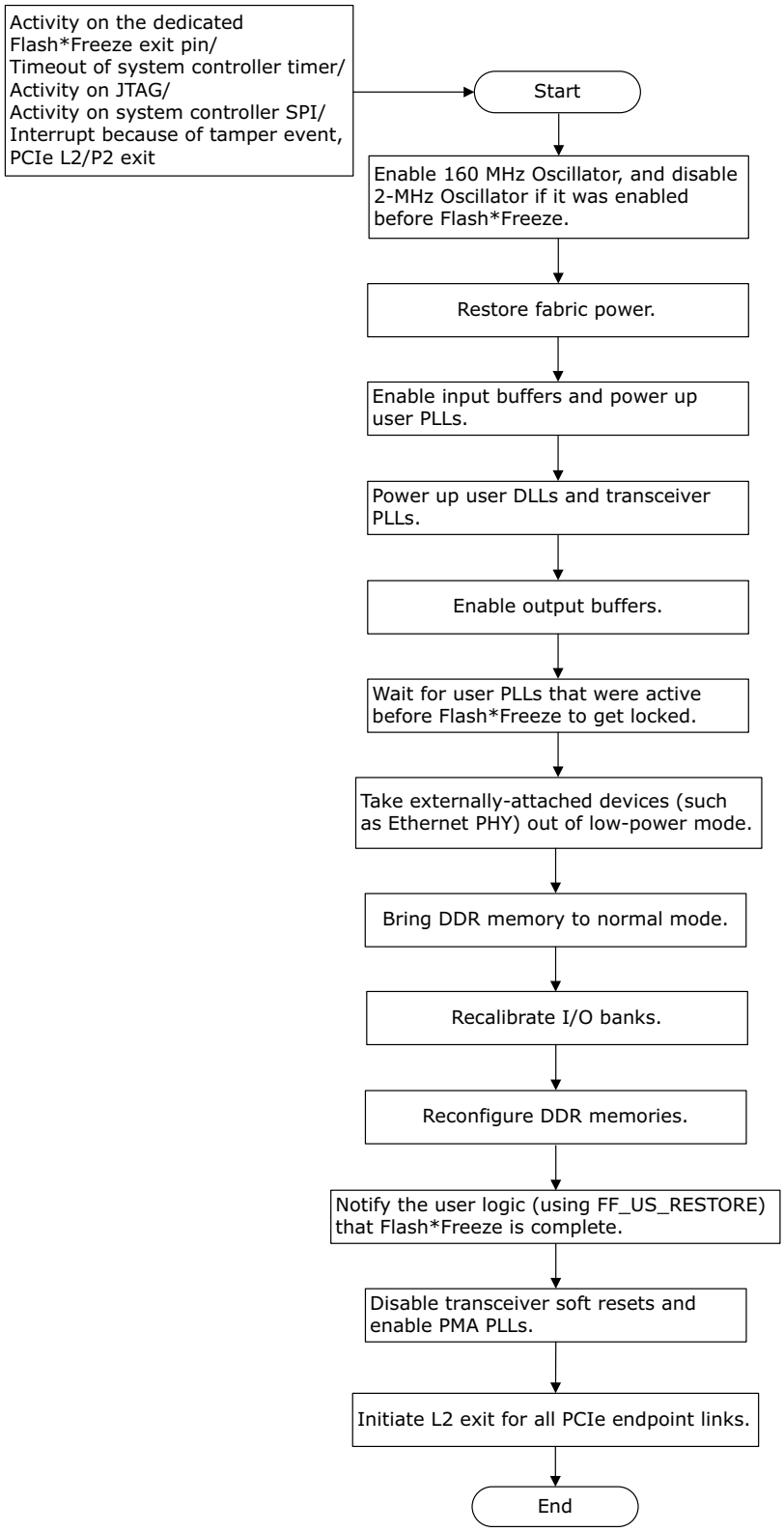
### 2.2.1.3 Flash*Freeze Exit

Flash*Freeze exit, which returns the device to normal operation, is initiated through any one of the following methods:

- Activity on the dedicated Flash*Freeze exit pin (FF_EXIT_N).
- Expiration of the Flash*Freeze duration mentioned in the CoreSysServices_PF IP configurator.
- Activity on the JTAG interface.
- Activity on the SPI SSEL pin of the system controller.
- Interrupt to the system controller from analog detectors (tamper event) or because of PCIe L2/P2 exit. For more information, see Host-Initiated L2 Exit, page 11.

The following digram shows the sequence of steps executed by the system controller for exiting Flash*Freeze.

***Figure 7 •*** **Flash*Freeze Exit**



#### 2.2.1.3.1 FF_US_RESTORE

When the fabric comes out of Flash*Freeze, the flip-flops in the fabric are restored to their previous values. During this process, a glitch may occur, which may impact the state of the flip-flops. The glitch only impacts flip-flops fed by an asynchronous set or reset signal of another flip-flop. The FF_US_RESTORE signal from the CoreSysServices_PF IP must be used by the fabric logic to gate off any such resets. If a flip-flop in the fabric is fed by a synchronous set or reset signal, the glitch does not affect the state of the downstream flip-flop, as the clocks are not restarted.

## 2.2.2 Transceiver PMA/PCS Low-Power Modes

The transceiver PMA/PCS is provided with three low-power modes—P0s, P1, and P2. These three low-power modes signify different levels of clock gating during Flash*Freeze. These low-power modes can be set at the design capture stage using the static flash configuration bits or the **Device Configuration and Memory Initialization** option in Libero SoC PolarFire. For more information about the transceiver PMA/PCS low-power mode, see *UG0677: PolarFire FPGA Transceiver User Guide*.

## 2.2.3 PCIe Low-Power Modes

Using P0s, P1, and P2 low-power modes, the PCIe link can be kept partially active, even when the fabric is powered down. The register settings available in the PCIe subsystem allow different levels of clock gating during Flash*Freeze. For more information about levels of clock gating, see Figure 5, page 6.

PCIe low-power modes do not use the optional auxiliary power specified in the PCIe standard. All blocks in a PolarFire device (other than the FPGA fabric) remain powered up during Flash*Freeze.

### 2.2.3.1 PCIe Link State L1

The PCIe link may enter L1 power-saving state from the normal L0 state as a result of either hardware-controlled active state power management (ASPM) or host software-controlled change of the link state. This is taken care of within the PCIe controller hardware and transceiver PMA/PCS. However, support is not provided in PolarFire FPGAs to relate Flash*Freeze with the L1 power-saving state.

### 2.2.3.2 PCIe Link State L2

The PCIe link on an endpoint may enter L2 power-saving sate from the normal L0 state as a result of a change of power state controlled by the host software. L2 is a lower-power state than L1 and has longer exit latency.

### 2.2.3.3 PCIe Low-Power Modes and Flash*Freeze

A PolarFire device can be configured as either a PCIe root port or a PCIe endpoint. If configured as a PCIe endpoint, upon entering PCIe L2 state, the device can be placed in Flash*Freeze. However, Flash*Freeze is independent of PCIe L2 state; it is not mandatory that the device be placed in Flash*Freeze during L2 state. If the device is configured as a PCIe root port, then Flash*Freeze is not supported during L2 state.

For an endpoint in L2 state, the trigger to exit L2 state may come from a local event at the endpoint itself or from the host (root port).

### 2.2.3.4 L2 Entry

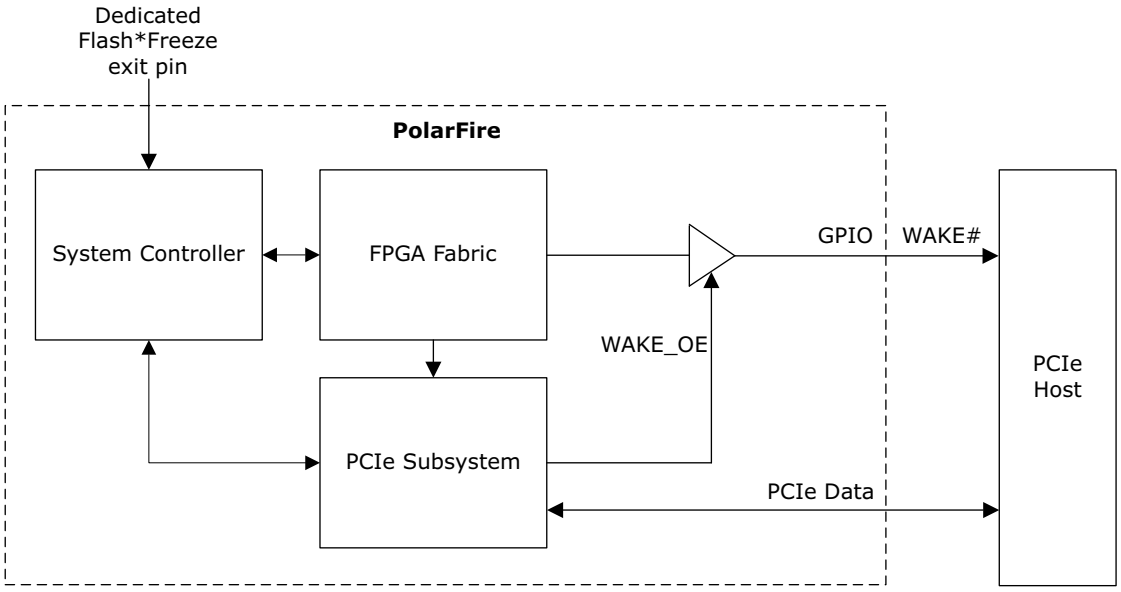
When the PCIe links enter L2 state, the LTSSM[4:0] register is used by the CoreSysServices_PF IP to indicate the PCIe links are in low-power state. The LTSSM[4:0] register is clocked using the PHY clock from the PCIe controller to the fabric. The CoreSysServices_PF IP can use this status to initiate Flash*Freeze entry, if required.

### 2.2.3.5 Endpoint-Initiated L2 Exit

If there is a local event at the endpoint, which requires the endpoint to initiate L2 exit, then the device must exit Flash*Freeze before waking up the PCIe link. The device may use either the out-of-band WAKE# mechanism or an in-band beacon to the host to exit low-power mode.

In the following illustration, the WAKE# signal is assigned to a GPIO. The endpoint event causes Flash*Freeze exit, which then powers up the fabric and allows the WAKE# signal to be sent to the host.

***Figure 8 •*** **Endpoint-Initiated Flash*Freeze/L2 Exit**



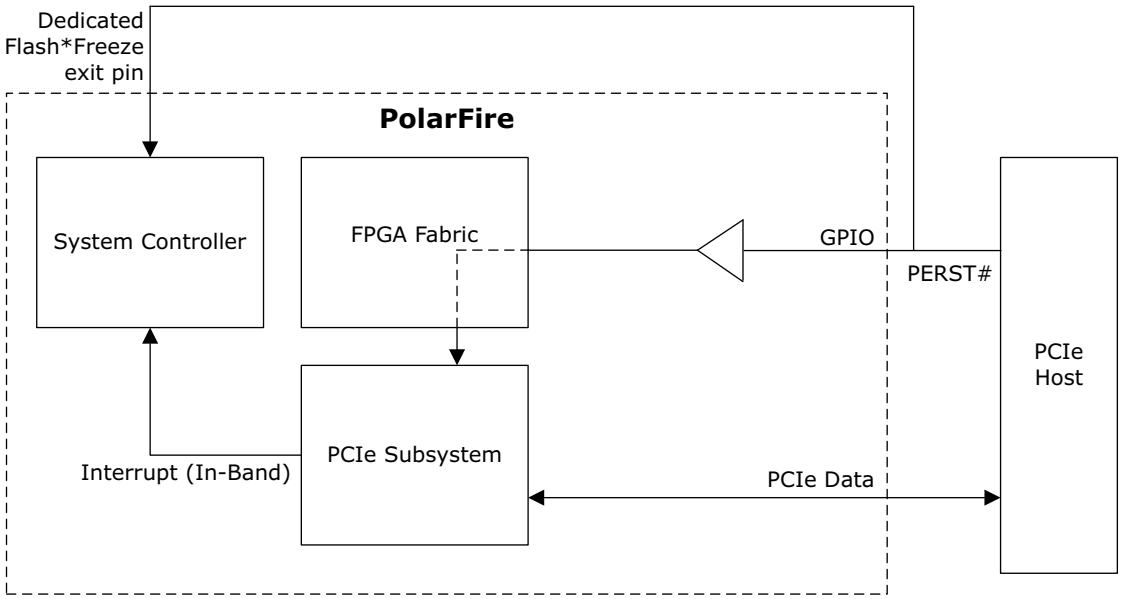

The device is forced to exit Flash*Freeze because of a local event caused by one of the supported mechanisms, for example, activity on the dedicated Flash*Freeze exit pin.

The fabric then powers up, and a signal is sent from the fabric to the PCIe subsystem informing it to begin the wake-up procedure. The PCIe subsystem either asserts the WAKE_OE signal, which helps to propagate the WAKE# signal to the host through the (repowered) fabric and the GPIO, or sends a beacon (if using in-band signaling).

### 2.2.3.6 Host-Initiated L2 Exit

If there is a host-initiated trigger (such as a break in electrical idle or the assertion of fundamental reset PERST#) to exit L2 state, then this trigger may be used to first trigger a Flash*Freeze exit at the endpoint and subsequently bring PCIe out of L2 state. The following figure shows how PERST# can be connected to trigger the device to exit Flash*Freeze. The PCIe subsystem sees PERST# and acts upon it to exit L2.

***Figure 9 •*** **Host-Initiated Flash*Freeze/L2 Exit**




The following are possible scenarios for Flash*Freeze exit:

- The PERST# signal is connected to the dedicated Flash*Freeze exit pin. When the host asserts PERST#, it causes Flash*Freeze exit. This powers up the fabric, which allows PERST# to flow through the fabric to the PCIe subsystem, causing it to reset.
- An in-band mechanism (such as a break in electrical idle) causes an interrupt to the system controller, which causes Flash*Freeze exit.

### 2.2.3.7 Out-of-Band L2 Exit

If the out-of-band PERST# signal is being used by the host to initiate L2 exit of the endpoint, then this can be fed into both the dedicated Flash*Freeze exit pin of the system controller and a GPIO. The GPIO is connected to the PCIe subsystem through the fabric to the PERST# input of that block. Both the dedicated Flash*Freeze exit pin and the GPIO assigned to PERST# must be powered by either a 2.5 V or 3.3 V supply for the correct logic levels to be seen at the inputs (this is a 3.3 V signal). The 1.8 V power option is not supported for these banks when using this configuration.

When PERST# is asserted (low), the resultant activity on the Flash*Freeze dedicated exit pin causes the system controller to initiate Flash*Freeze exit. When Flash*Freeze exit is complete, the fabric is powered up, and the PERST# input reaches the PCIe subsystem block, which then gets powered up.

### 2.2.3.8 In-Band L2 Exit

An in-band mechanism (such as a break in electrical idle) can be used by the host to trigger L2 exit of the endpoint and send an interrupt to the system controller, triggering Flash*Freeze exit. After the fabric is powered, the CoreSysServices_PF IP in the fabric can communicate with the PCIe subsystem to allow PCIe link operation to be restarted.

## 2.2.4 DDR Controller Low-Power Modes

The PolarFire FPGA DDR subsystem supports the following low-power operating modes:

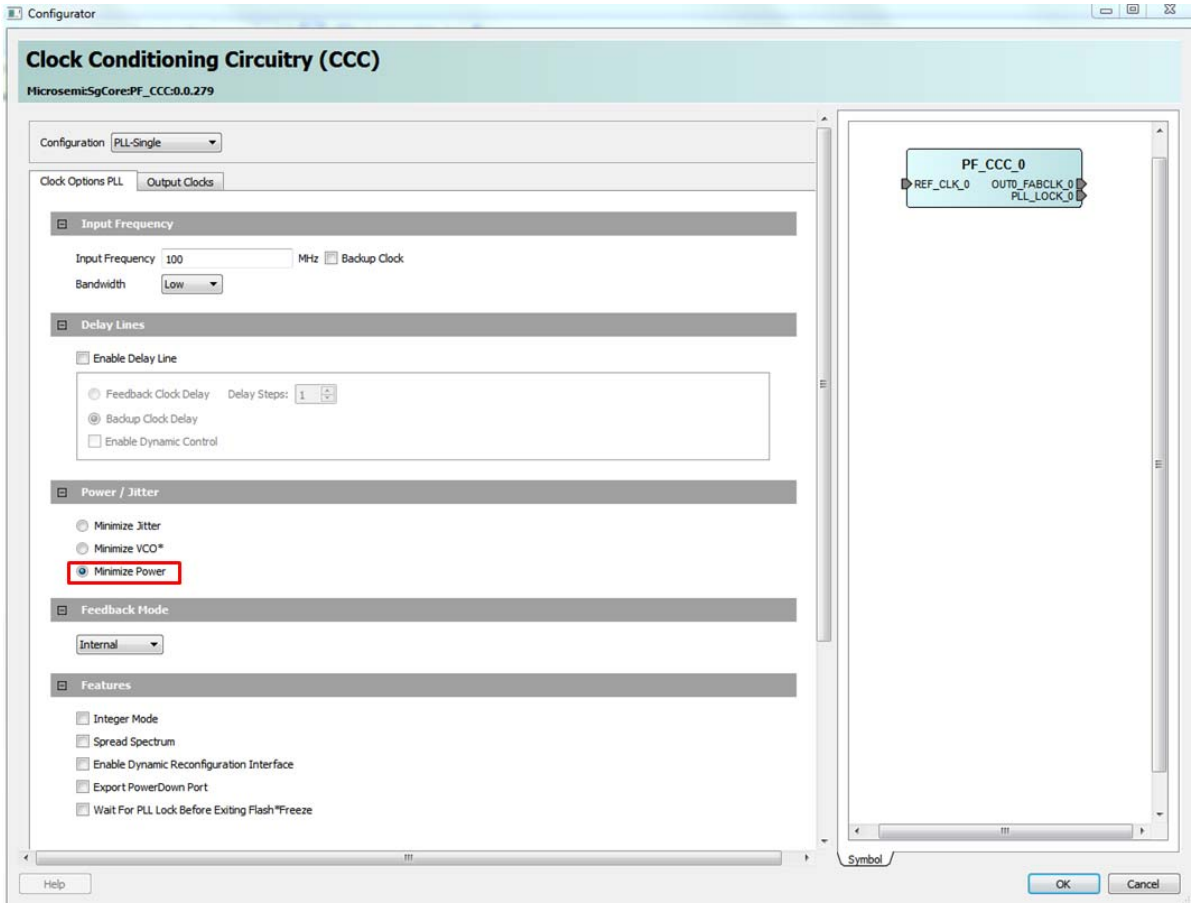
- Self-refresh mode: deactivates the clock to reduce power consumption of the device. This mode is similar to the standby mode where data is not lost.
- Power-down mode: puts the DDR memory in low-power mode and issues refresh commands automatically to retain data.

For more information about DDR low-power modes, see *UG0676: PolarFire FPGA DDR Memory Controller User Guide*.

## 2.2.5 CCC and RAM Blocks Low-Power Mode

The voltage-controlled oscillator (VCO) can be set for minimum range at the cost of higher jitter by selecting **Minimize Power**, as shown in the following figure.

***Figure 10 •*** **Setting to Minimize Power in CCC Configurator**




For more information about the CCC configurator, see *UG0684: PolarFire FPGA Clocking Resources User Guide*.

The dual-port LSRAM, two-port LSRAM, and micro SRAM can also be set for minimum power consumption. This is done through additional logic in the input and output using depth cascading. The following figures show how the dual-port LSRAM, two-port LSRAM, and micro SRAM are set to low power.

***Figure 11 •*** **Low-Power Setting for Dual-Port LSRAM in CCC Configurator**

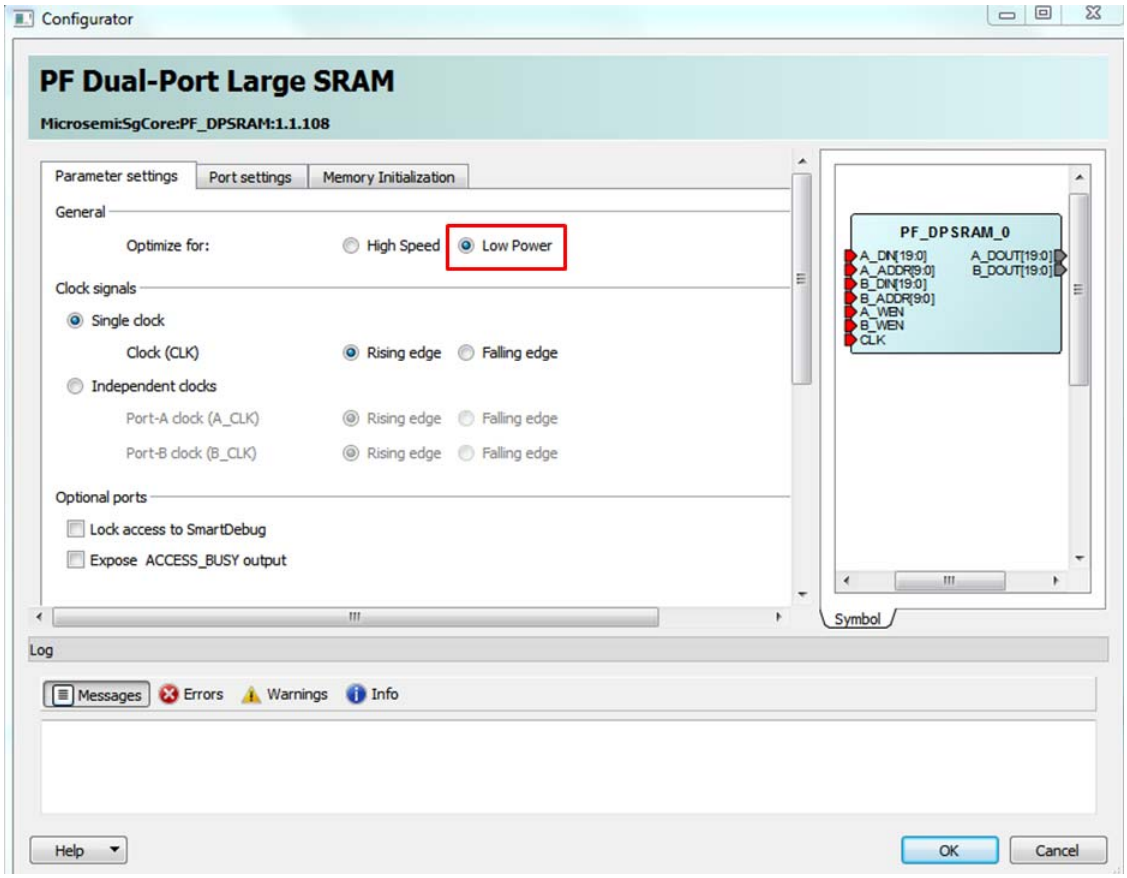



***Figure 12 •*** **Low-Power Setting for Two-Port LSRAM in CCC Configurator**

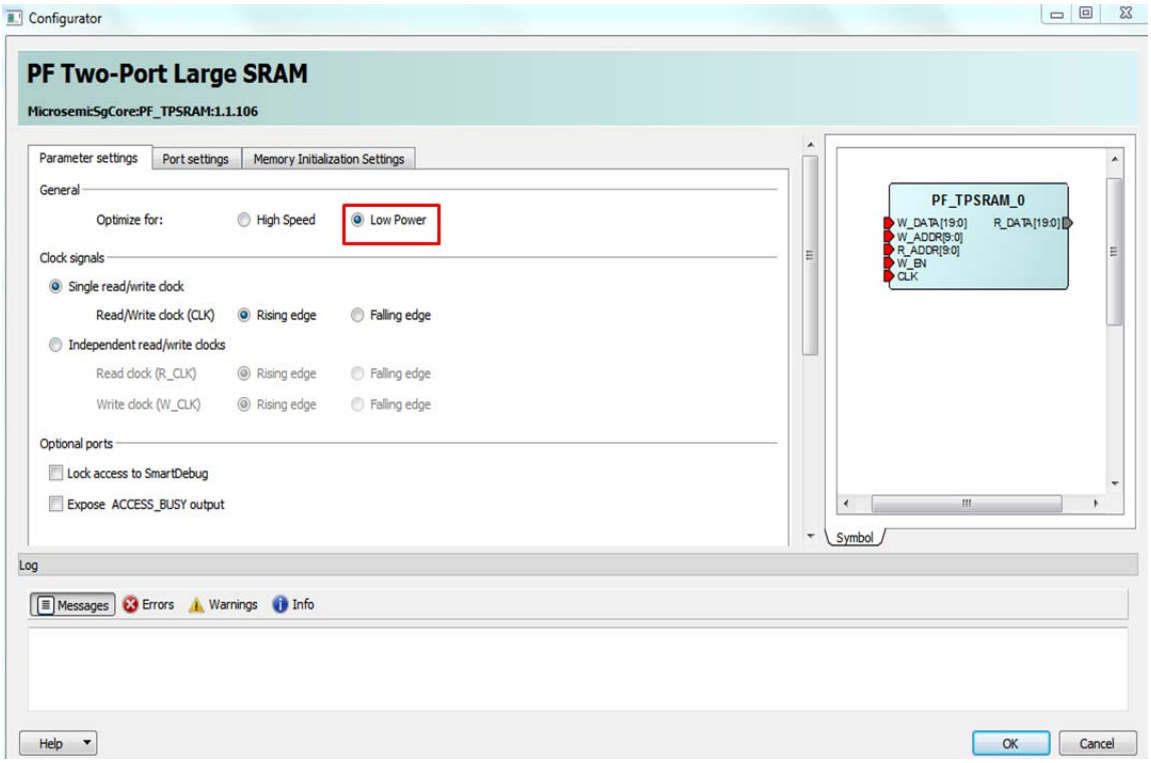
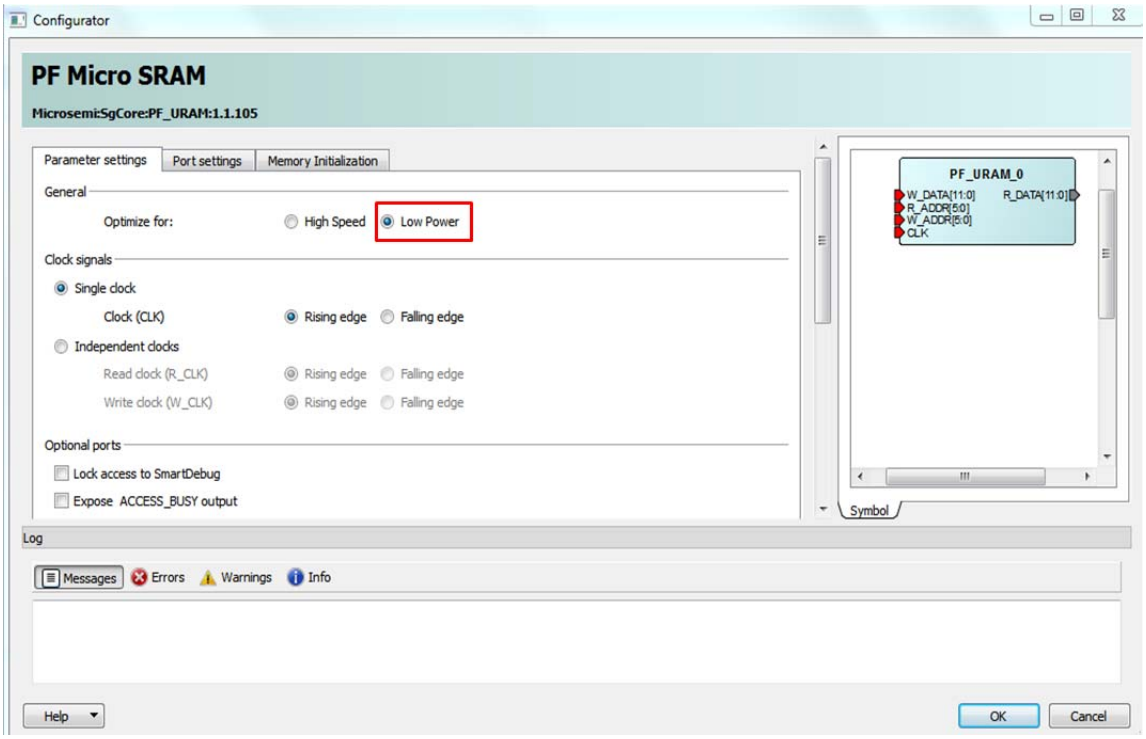


***Figure 13 •*** **Low-Power Setting for Micro SRAM in CCC Configurator**



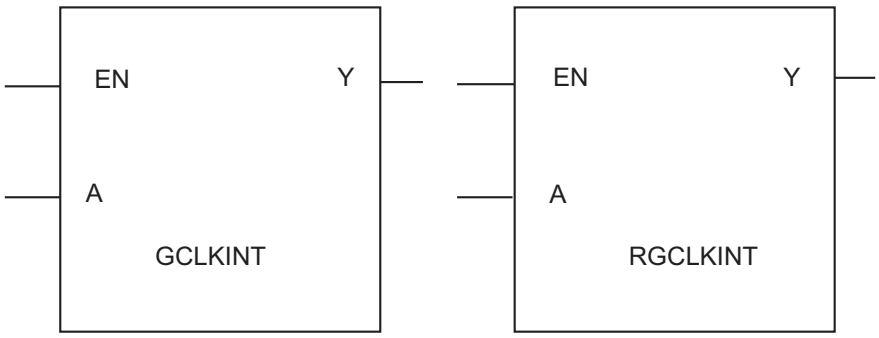

## 2.3 Clock Gating for Flash*Freeze

Microsemi recommends using a synchronous design, cleanly gating all the internal and external clocks. This prevents narrow pulses upon entry to and exit from Flash*Freeze.

PolarFire devices include sophisticated clock gating for the entire global clock network or for portions of it. All library macros referencing global clock drivers include an enable input to gate the clock when required. The following illustration shows the global clock macros with the clock gating feature. The clock is passed when the enable (EN) pin is asserted high, and is held low when the EN pin is disabled. A portion of a clock network can be clock gated using the RGCLKINT macro.

Similar to a flip-flop, the EN signal must be checked for basic setup and hold-timing violations to prevent glitches.

***Figure 14 •*** **Global Gated Clock Macros**




The following illustrations show example designs that use single and dual global gated clock buffer macros to control the fabric user logic.

***Figure 15 •*** **Gated External Clock Using Single GCLKINT Macro**

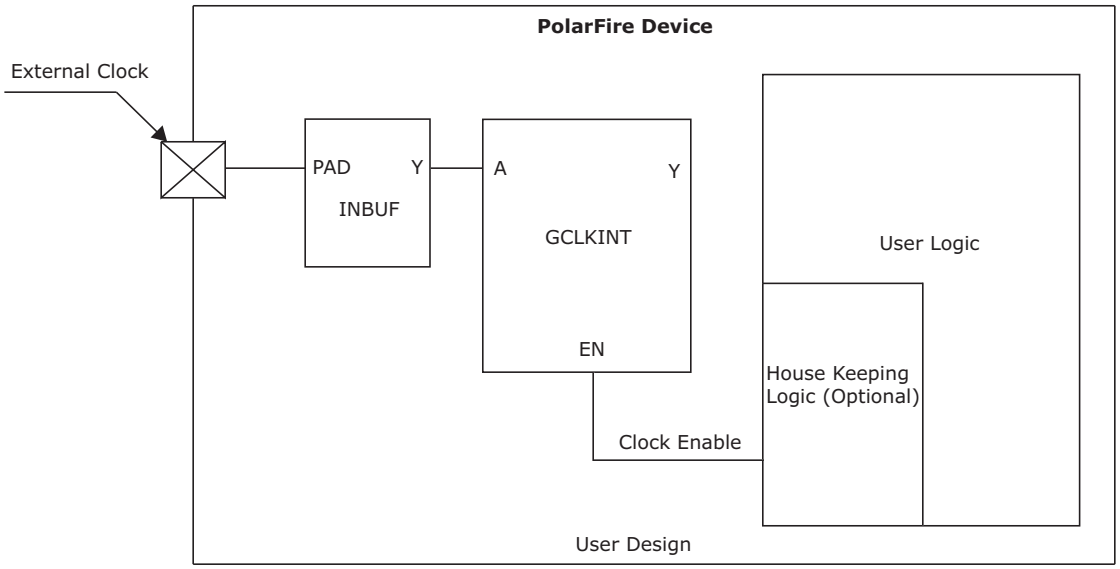
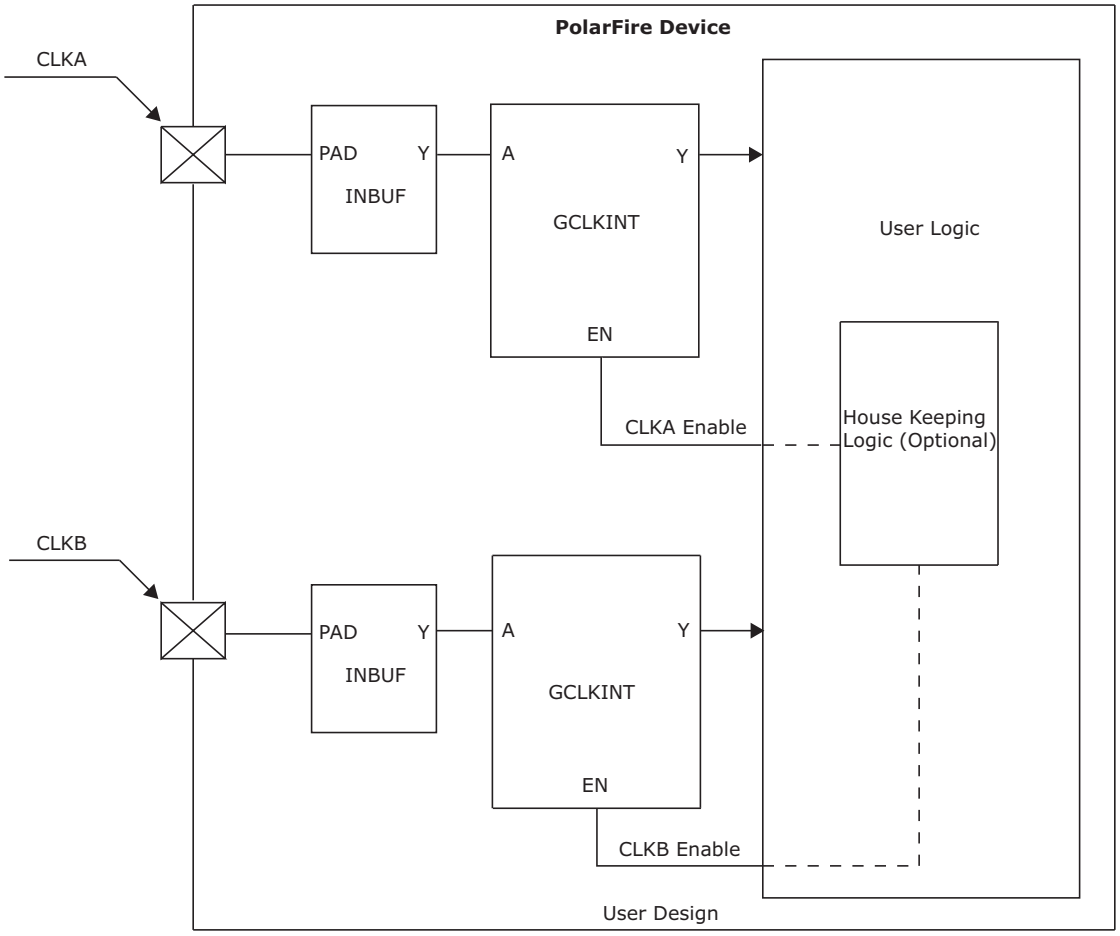



***Figure 16 •*** **Gated External Clock Using Multiple GCLKINT Macros**




The clocks can continue to drive the FPGA pins while the device is in Flash*Freeze, with virtually no power consumption. After exiting Flash*Freeze, the design must allow maximum acquisition time for the PLL to acquire the lock signal and for PLL clock activation. If you choose to wait for the PLL lock before exiting Flash*Freeze (as shown in the preceding illustration), the lock time is included in the

Flash*Freeze duration. The clocks at the CCC output can be gated to achieve low power. For more information on how to perform clock gating on CCC output clocks, see *UG0684: PolarFire FPGA Clocking Resources User Guide*.

## 2.4 Use Models

Microsemi supports the following three Flash*Freeze use models for PolarFire devices:

- Fabric-Only
- Fabric+DDR
- Fabric+PCIe

### 2.4.1 Fabric-Only Use Model

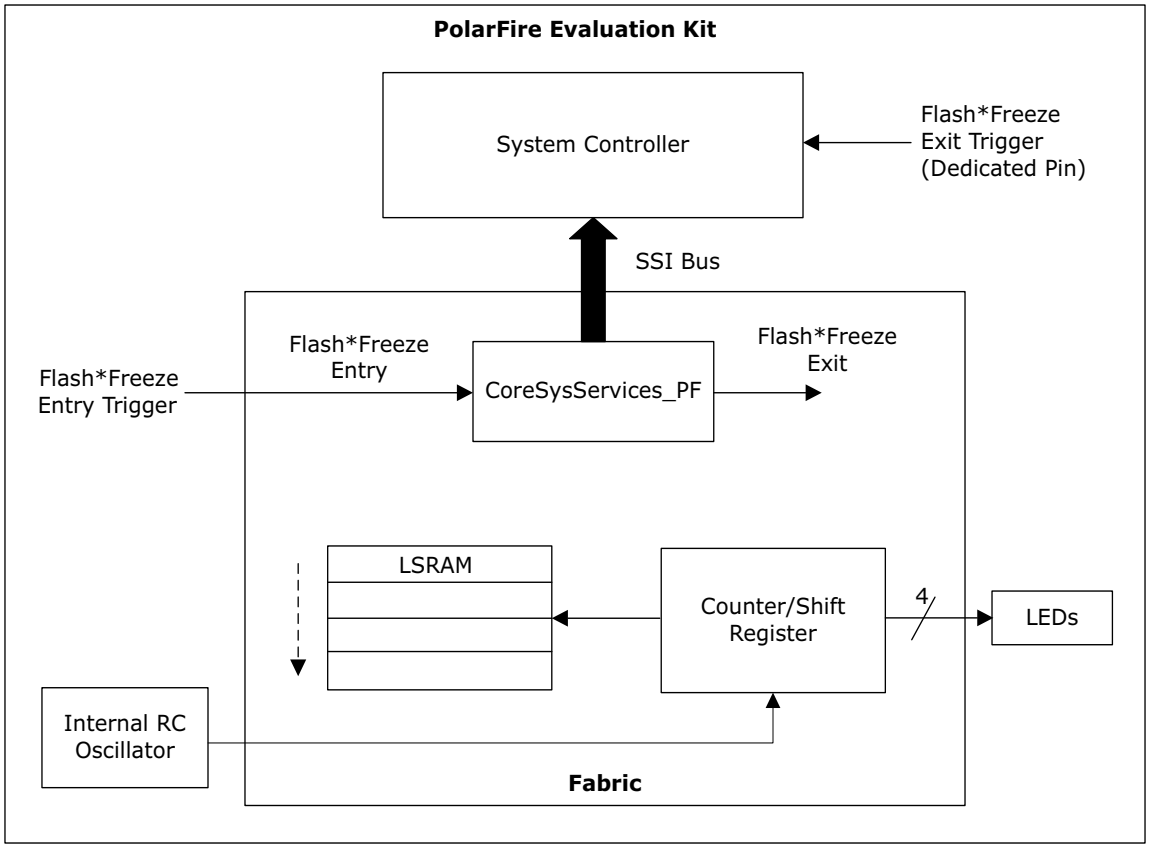
In this use model, the FPGA fabric (not PLL, DDR, or transceiver) is used to enter and exit Flash*Freeze.

The fabric logic has a counter that drives a set of LEDs that indicate the state of the fabric during Flash*Freeze. To demonstrate the state retention capability of LSRAM during Flash*Freeze, the counter/shift register data is loaded before Flash*Freeze entry and read for validity after Flash*Freeze exit.

The CoreSysServices_PF IP communicates with the system controller and initiates the Flash*Freeze system service. It also provides Flash*Freeze exit indication to the fabric.

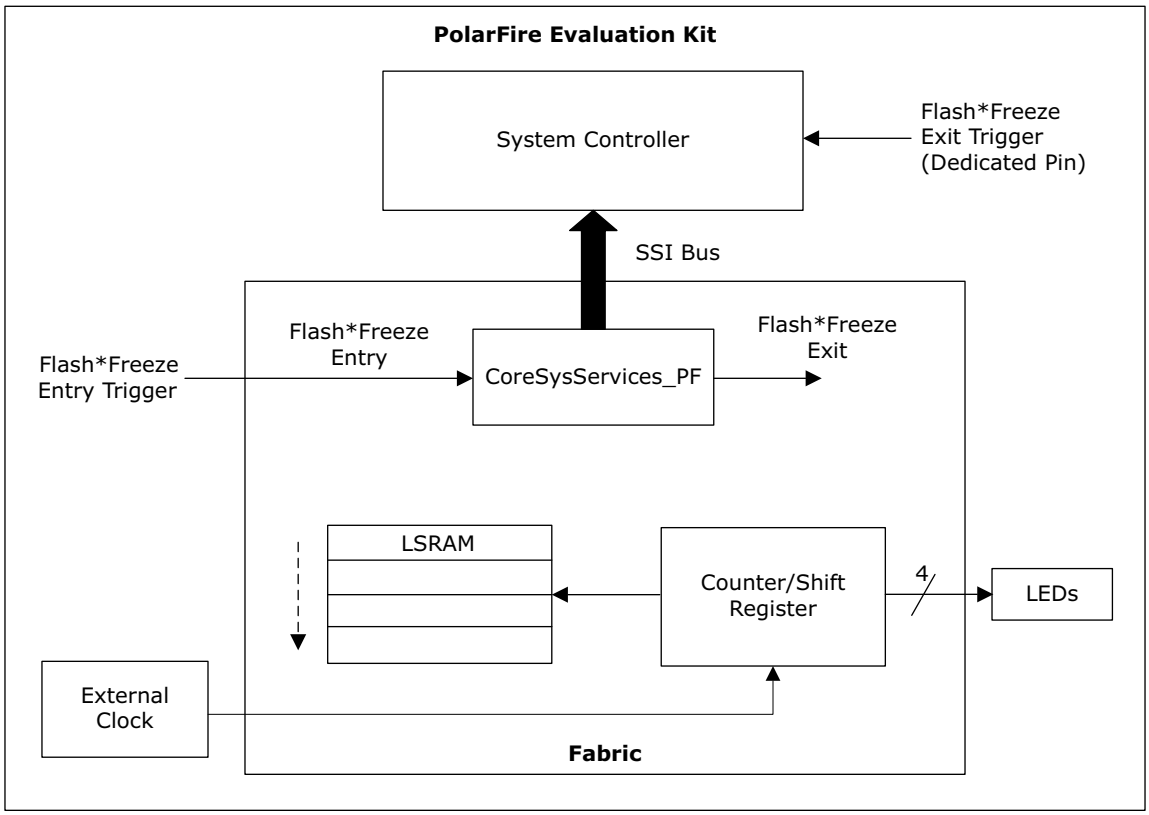
The following illustration shows the fabric-only use model using an internal RC oscillator.

***Figure 17 •*** **Fabric-Only Use Model Using Internal RC Oscillator**



The following illustration shows the fabric-only use model using an external clock.

***Figure 18 •*** **Fabric-Only Use Model Using External RC Oscillator**




## 2.4.2 Fabric+DDR Use Model

In this use model, the fabric, CCC, and DDR memory are used to enter and exit Flash*Freeze. The external DDR memory is placed in self-refresh mode before Flash*Freeze entry. The CCC receives an external clock as input and feeds the output to the DDR memory for read/write operations. The system controller waits for the CCC to lock before initiating Flash*Freeze exit.

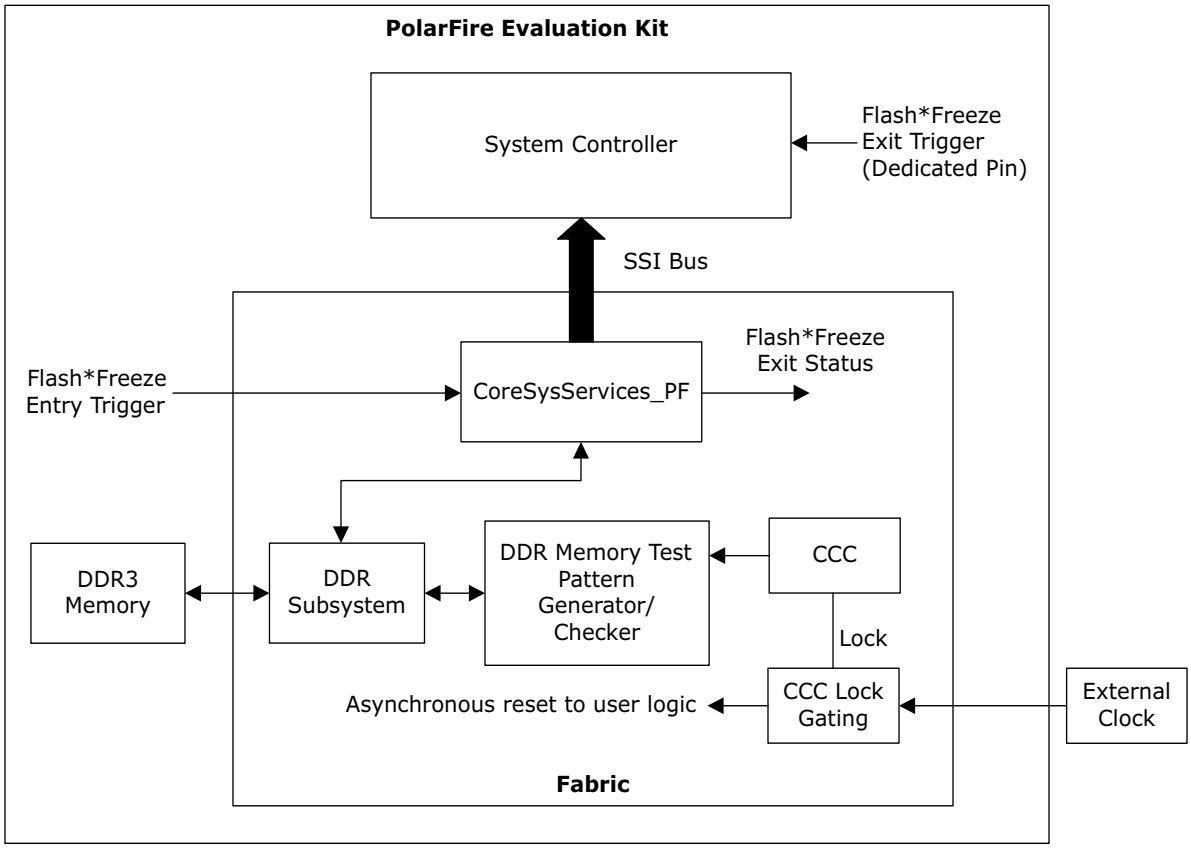
CoreSysServices_PF IP along with the DDR subsystem performs the following sequence of actions:

1. Communicates with the system controller to initiate and support Flash*Freeze service.
2. Before Flash*Freeze entry, places the external DDR3 memory in self-refresh mode.
3. Initiates Flash*Freeze service.
4. After Flash*Freeze exit, performs DDR retraining/recalibration.
5. Places the external DDR3 memory back to normal operation.
6. Provides Flash*Freeze exit status to the fabric.

**Note:** The CCC lock needs to be gated to avoid glitches before it can be used as an asynchronous reset to the fabric logic during Flash*Freeze exit. The signal used to reset the user logic can be gated with the CCC lock signal, the FF_US_RESTORE signal, and the INIT_DONE signal from the INIT macro. For more information about the INIT_DONE signal, see *UG0725: PolarFire FPGA Device Power-Up and Resets User Guide*.

The following illustration shows the fabric+DDR use model.

***Figure 19 •*** **Fabric+DDR Use Model**




## 2.4.3 Fabric+PCIe Use Model

In this use model, the fabric, CCC, and PCIe interfaces are used to enter and exit Flash*Freeze. The CCC receives an external clock as input and feeds the output to the PCIe interface for read/write operations. To differentiate between the active mode and the Flash*Freeze capability of PCIe, data is loaded to the SRAM through the AXI interconnect prior to Flash*Freeze entry and read for validity after Flash*Freeze exit. User PLLs that receive reference or feedback clocks from the fabric are in internal loopback mode.

The CoreSysServices_PF IP performs the following sequence of actions:

**Flash*Freeze Entry**

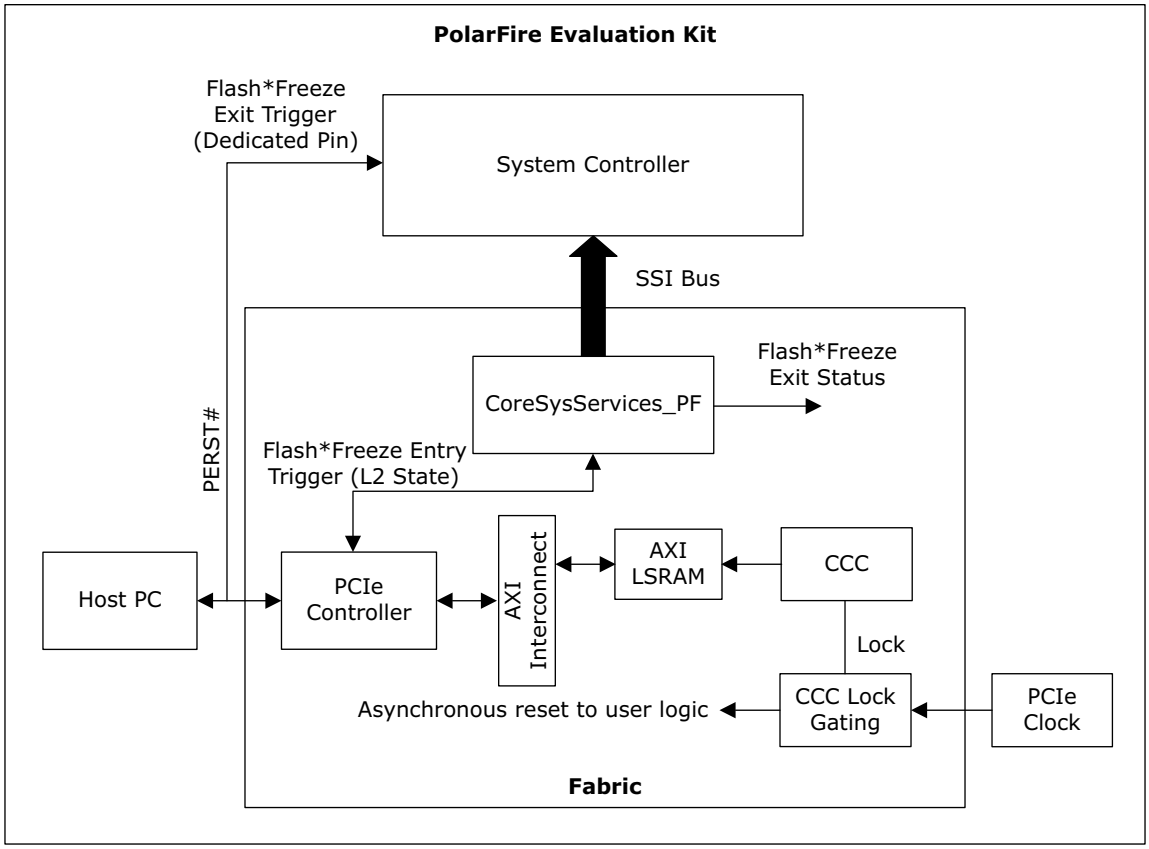
1. Detect the PCIe L2 low-power state (entry) initiated by the host and start the Flash*Freeze system service.
2. Look for LTSSM states to detect L2P2.

**Flash*Freeze Exit**

1. Connect PERST# to a dedicated Flash*Freeze exit pin on the PolarFire device.
2. Initiate Flash*Freeze exit upon PERST# assertion by the host.

The following illustration shows the fabric+PCIe use model.

***Figure 20 •*** **Fabric+PCIe Use Model**




## 2.4.4 Using Flash*Freeze in Libero SoC PolarFire

TBD