IGLOO 2 DDR Controller and Serial High-Speed Controller Initialization Methodology

Introduction (Ask a Question)

When designing with an IGLOO® 2 device, it is necessary to initialize the configuration registers of FDDR or MDDR controllers or serial high-speed controller (SERDESIF) blocks at runtime before they can be used. For instance, when using a DDR controller, it is important to set the DDR mode (DDR3/DDR2/LPDDR), PHY width, burst mode, and ECC. Similarly, for the SERDESIF block used as a PCIe endpoint, it is essential to set the PCIE BAR to an AXI or AHB window.

This guide describes how to create Libero® designs that initialize FDDR and MDDR controllers and SERDESIF blocks automatically at power-up.

Important: To initialize the DDR or SERDESIF blocks in your design, use the IGLOO 2 System Builder. System Builder addresses the creation of the configuration data for DDR controller and SERDESIF configuration registers, as well as the creation of the FPGA logic required to transfer the configuration data to different ASIC configuration registers.

For information about the DDR controller and SERDESIF configuration registers, see the SmartFusion 2 and IGLOO 2 FPGA High-Speed Serial Interface User Guide and SmartFusion 2 and IGLOO 2 FPGA High-Speed DDR Interfaces User Guide.

Important: This document uses “Master” and “Slave.” The equivalent Microchip terminology used in this document is **Initiator** and **Target**, respectively.

Table of Contents

1. Theory of Operation (Ask a Question)

The peripheral initialization solution consists of the following major components:

- The CoreConfigMaster soft IP core is an AHBLite master to the High Performace Memory Subsystem (HPMS) via FIC_0 (HPMS FIC_0-Fabric Master Subsystem) and directs the initialization process.
- The CoreConfigP soft IP core initializes the peripherals' configuration registers.
- The CoreResetP soft IP core handles the reset sequence of the HPMS, DDR controllers, and SERDESIF blocks.
- The peripheral initialization data is stored in the HPMS eNVM by Libero and is fetched by the CoreConfigMaster in runtime to configure the peripheral configuration registers.

When programming the IGLOO 2 device with an IGLOO 2 design that utilizes either the DDR controllers or the SERDESIF blocks, Libero programs the reserved pages of the HPMS eNVM with peripheral initialization data in hexadecimal format. It is necessary to configure or load the configuration registers of the DDR controllers and SERDESIF blocks with this initialization data stored in the HPMS eNVM.

The following steps describe the peripheral initialization process:

1. During reset, the CoreResetP output signal MSS_HPMS_READY is asserted at the beginning of the initialization process. This indicates that the HPMS block, except MDDR, is ready to communicate.
   The CoreConfigMaster soft IP core, functioning as an AHBLite master to the HPMS via FIC_0 in the HPMS FIC_0 Fabric Master Subsystem, is then activated and commences the initialization process.
2. The peripheral configuration data stored in the eNVM is fetched by the CoreConfigMaster that is mastering the HPMS via FIC_0, and is loaded into the configuration registers of the FDDR/MDDR controllers and SERDESIF via the HPMS FIC_2 APB3 interface. This HPMS FIC_2 interface is connected to the CoreConfigP soft IP core instantiated in the FPGA fabric.
3. After the registers are configured, the CoreConfigP output signals CONFIG1_DONE and CONIG2_DONE are asserted.
   There are two phases of register configuration, CONFIG1 and CONFIG2, depending on the peripherals used in the design.
4. If one or both of MDDR/FDDR are used, and none of the SERDESIF blocks is used in the design, then there is only one register configuration phase. Both the CoreConfigP output signals CONFIG1_DONE and CONIG2_DONE are asserted, one after the other, without any delay.
   If the design includes one or more SERDESIF blocks that are in non-PCIe mode, then there is also only one phase of register configuration. CONFIG1_DONE and CONIG2_DONE are asserted, one after the other, without any delay.
   If one or more SERDESIF blocks in PCIe mode are used in the design, then there are two phases of register configuration. CONFIG1_DONE is asserted after the first phase of register configuration is complete. SERDESIF system and lane registers are configured in this phase. If SERDESIF is configured in non-PCIe mode, then the CONFIG2_DONE signal is also asserted immediately.
5. If SERDESIF is configured in PCIe mode, then the second phase of register configuration follows. The following events occur in the second phase:
   - CoreResetP de-asserts PHY_RESET_N and CORE_RESET_N signals corresponding to each SERDESIF block used. It also asserts the output signal SDIF_RELEASED after all the SERDESIF blocks are out of reset. This SDIF_RELEASED signal indicates to the CoreConfigP that the SERDESIF core is out of reset and ready for the second phase of register configuration.

   - After the SDIF_RELEASED signal is asserted, polling for the assertion of PMA_READY on the appropriate SERDESIF lane starts. After the PMA_READY is asserted, the second set of SERDESIF (PCIe) registers are configured.
6. After all the PCIE registers are configured, the CoreConfigP control registers are configured to indicate the completion of the second phase of register configuration. The CoreConfigP output signal CONIG2_DONE is then asserted.
7. In addition to the above signal assertions or de-assertions, CoreResetP also manages the initialization of various blocks by performing the following functions:
   - De-asserting the FDDR core reset
   - De-asserting the SERDESIF blocks PHY and CORE resets
   - Monitoring of the FDDR PLL (FPLL) lock signal. The FPLL must have locked to guarantee that the FDDR AXI/AHBLite data interface and the FPGA fabric can communicate properly.
   - Monitoring of the SERDESIF block PLL (SPLL) lock signals. The SPLL must have locked to guarantee that the SERDESIF blocks AXI/AHBLite interface (PCIe mode) or XAUI interface can communicate properly with the FPGA fabric.
   - Waiting for the external DDR memories to settle and be ready to be accessed by the DDR controllers.
8. When all peripherals complete their initialization, CoreResetP asserts the INIT_DONE signal and the CoreConfigP internal register INIT_DONE is asserted.
   If one or both of MDDR/FDDR are used, and the DDR initialization time is reached, CoreResetP output signal DDR_READY is asserted. Assertion of DDR_READY signal DDR_READY can be monitored as an indication that the DDR (MDDR/FDDR) is ready for communication.
   If one or more SERDESIF blocks are used, and the second phase of register configuration is successfully completed, CoreResetP output signal, SDIF_READY, is asserted. Assertion of the SDIF_READY signal can be monitored as an indication that all the SERDESIF blocks are ready for communication.

The following three figures show the initialization steps of FDDR/MDDR, SerDes (non-PCIe mode), and SerDes (PCIe mode).

Figure 1-1. MDDR/FDDR (Without SERDESIF) Initialization Flow Chart

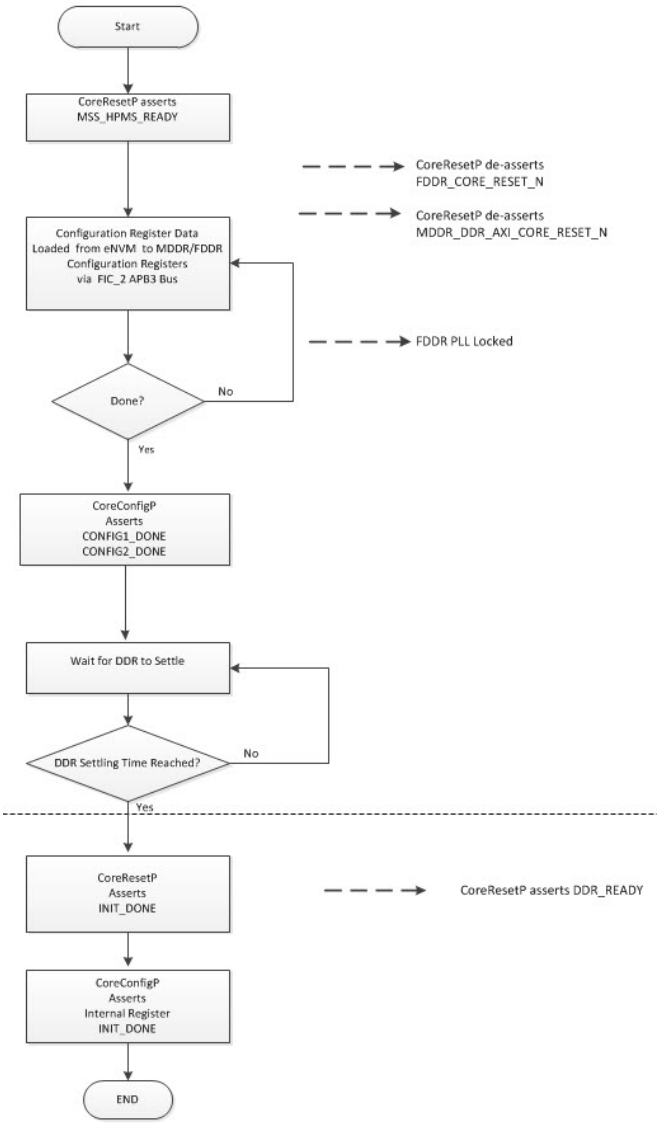


Figure 1-2. SERDESIF (Non-PCIe) Initialization Flow Chart

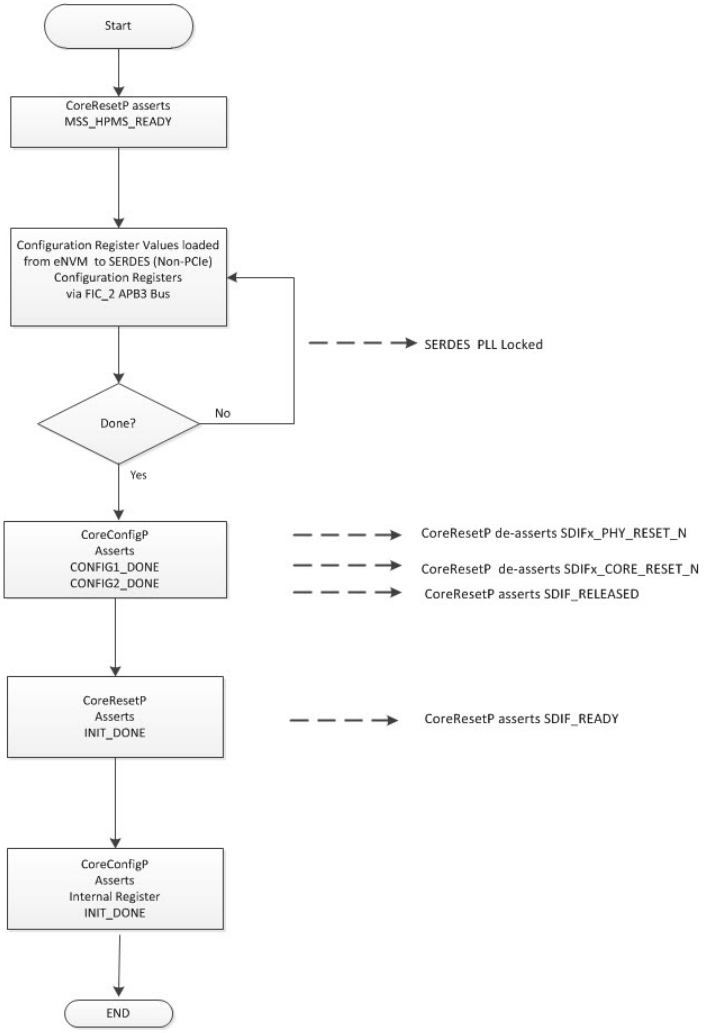


Figure 1-3. SERDESIF (PCIe) Initialization Flow Chart

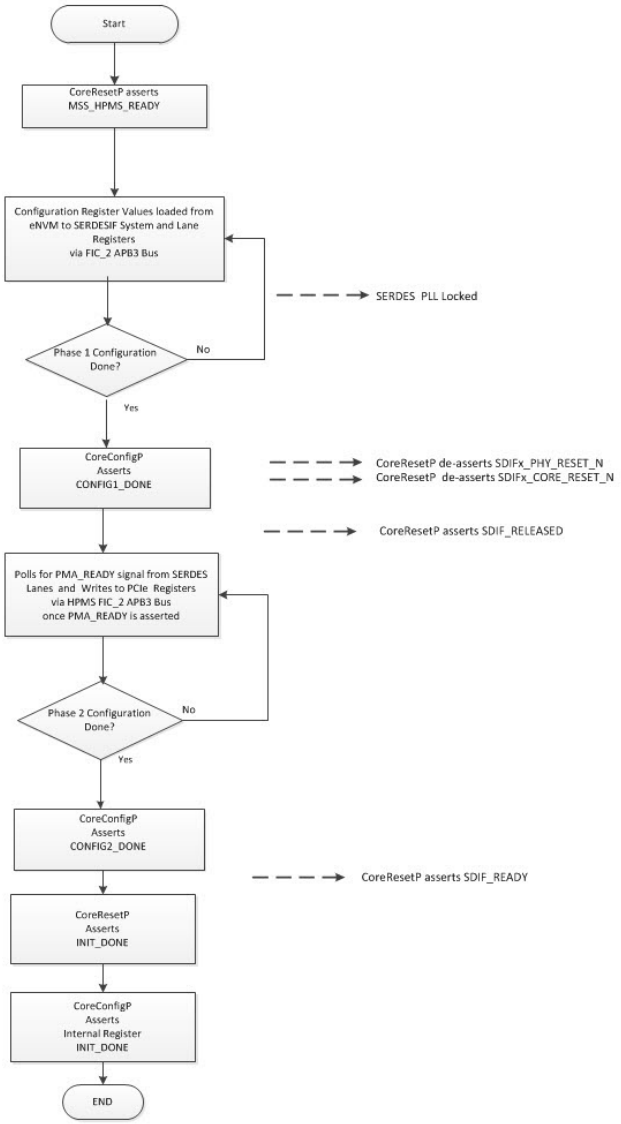
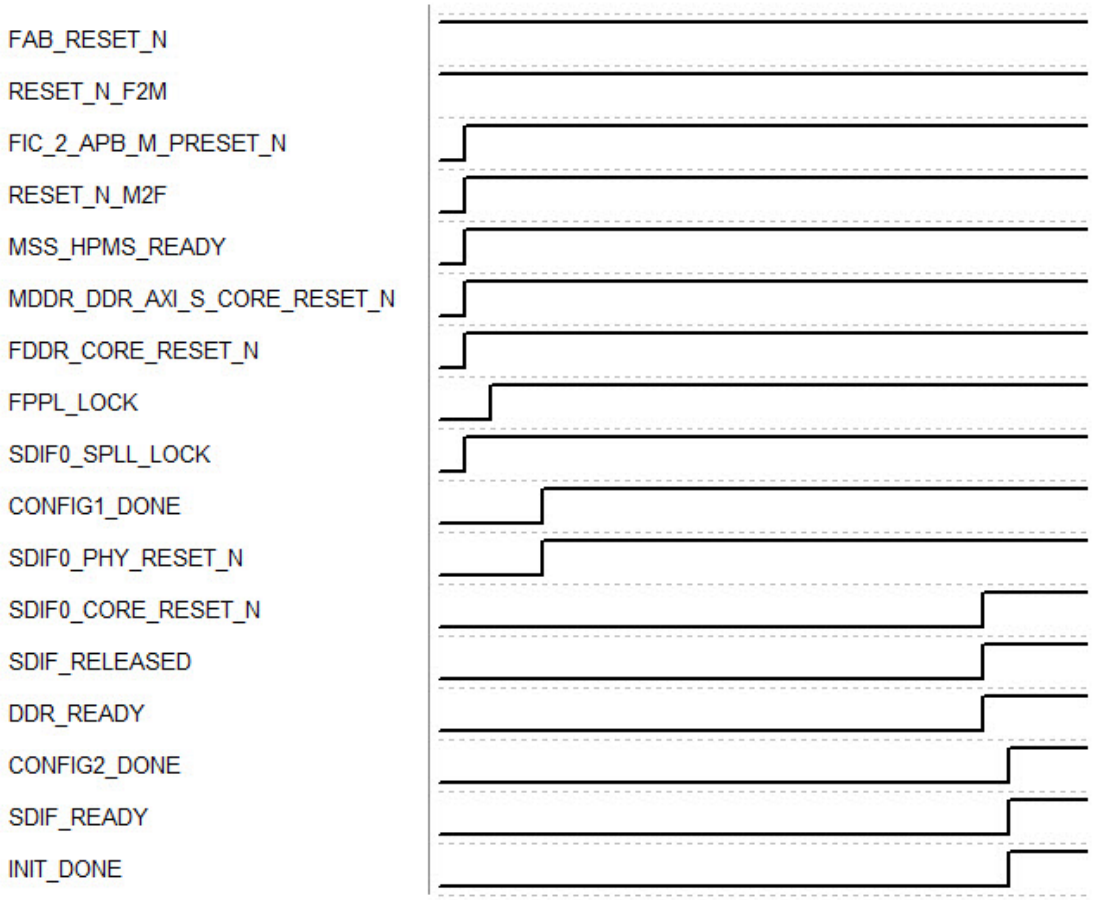


Figure 1-4. Peripheral Initialization Timing Diagram



2. Creating a Design (Ask a Question)

System Builder allows you to capture system-level requirements and produce a design that implements those requirements. To achieve this, System Builder creates the peripheral initialization subsystem automatically.

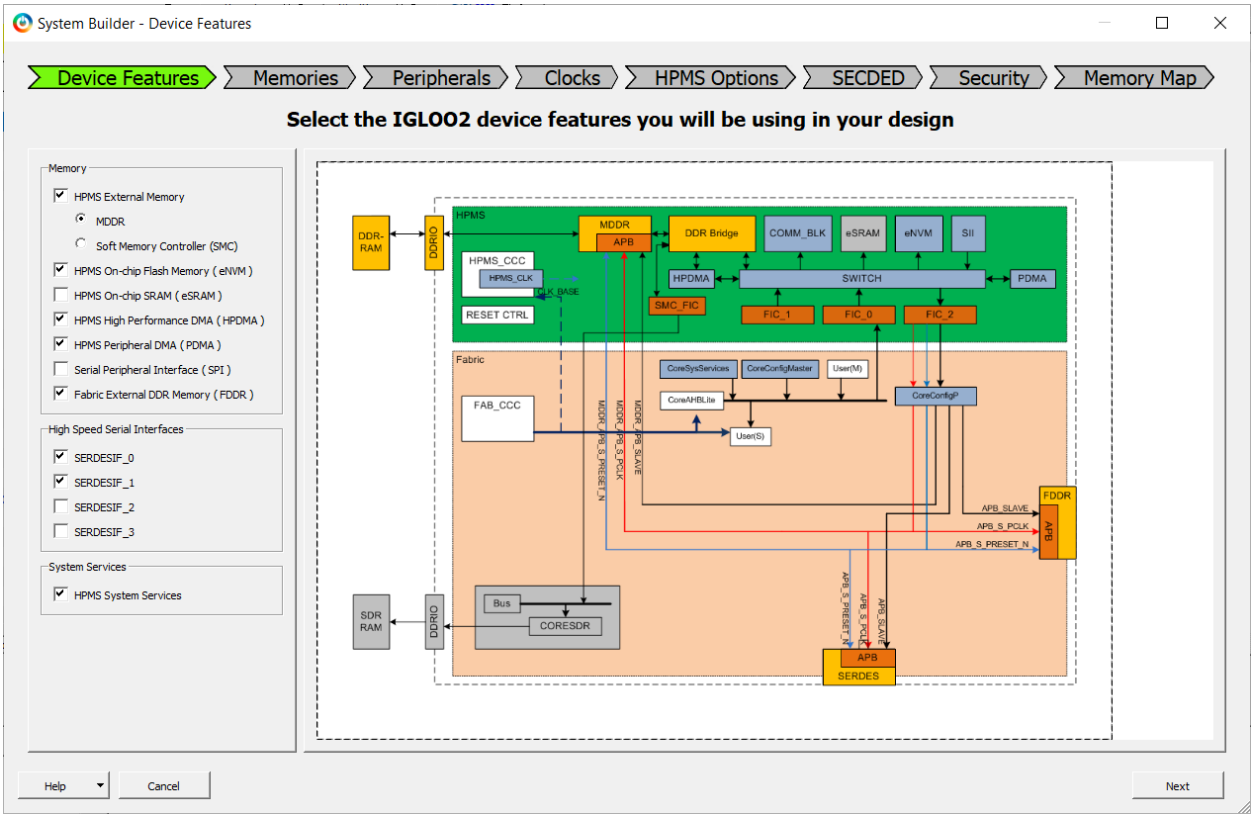
To create a design that initializes your DDR controllers and SERDESIF blocks at power-up, perform the following procedure:

1. In the Device Features page, specify which DDR controllers are used and how many SERDESIF blocks are used in your design.
   You can also select HPMS High Performance DMA (HPDMA) and HPMS eNVM or HPMS eSRAM to initiate and manage data transfers with the external DDR memory via HPMS MDDR at one end and eNVM/eSRAM/fabric peripherals at the other end.
2. In the Memories page, specify the type of DDR (DDR2/DDR3/LPDDR) and the configuration data for your external DDR memories.
3. In the Peripherals page, add fabric masters configured as AHBLite/AXI to the Fabric DDR subsystem, HPMS DDR FIC subsystem, or both (optional).
4. In the Clock Settings page, specify the clock frequencies for the DDR subsystems.
5. Complete your design specification and click **Finish**. This generates the System Builder created design, including the logic necessary for the initialization solution.
6. If you use SERDESIF blocks, instantiate the SERDESIF blocks in your design and connect their initialization ports (bus interface, clock and reset ports) to those of the System Builder generated block.

2.1 System Builder Device Features Page (Ask a Question)

In the System Builder Device Features page, you specify which DDR controllers (MDDR and/or FDDR) are used and how many SERDESIF blocks are used in your design. You can also select HPMS High Performance DMA (HPDMA) and HPMS eNVM or HPMS eSRAM.

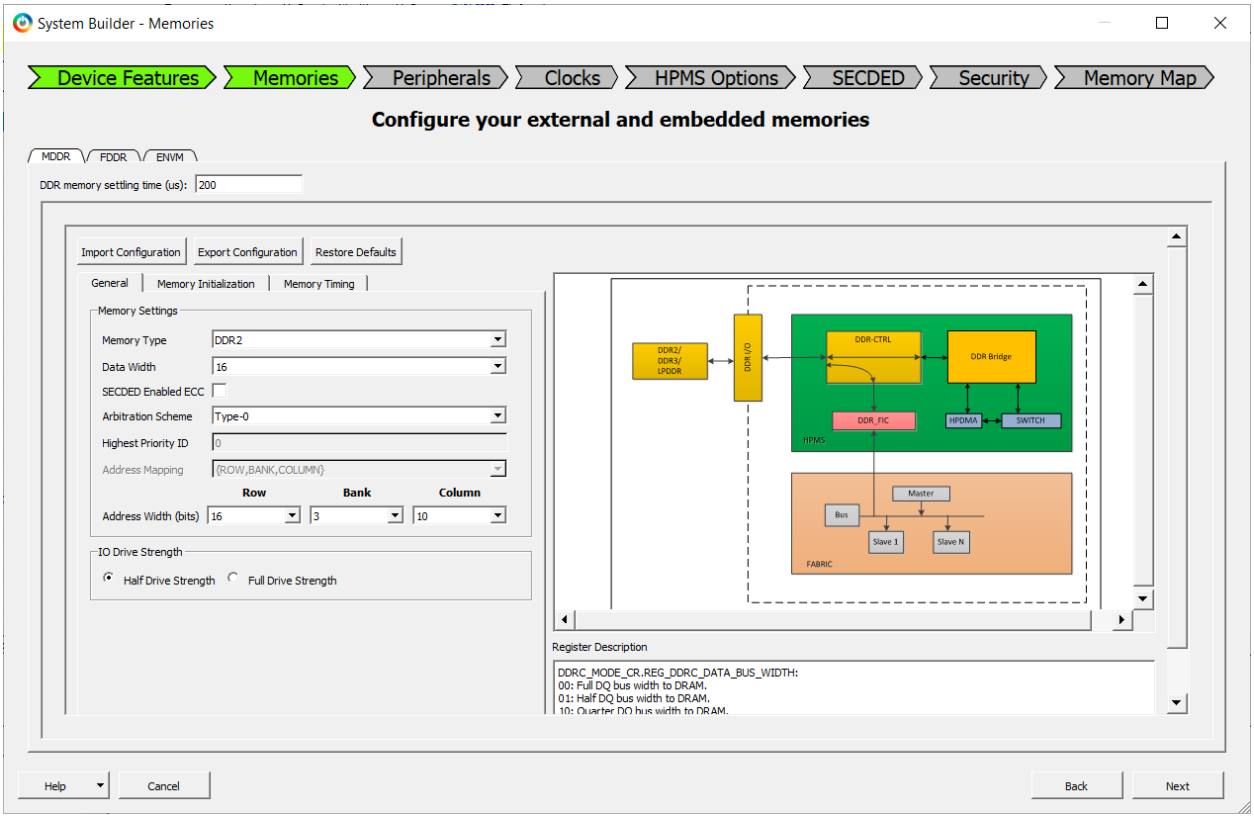
Figure 2-1. System Builder Device Features Page



2.2 System Builder Memories Page (Ask a Question)

To configure the HPMS DDR (MDDR) or Fabric DDR (FDDR), select the **MDDR** or **FDDR** tab in the System Builder Memories page.

Figure 2-2. HPMS External Memory



The HPMS DDR (MDDR) and Fabric DDR (FDDR) controllers must be configured dynamically at runtime to match the external DDR memory configuration requirements (DDR mode, PHY width, burst mode, ECC, and so on). Data entered in the **MDDR/FDDR** tabs in the System Builder Memories page is loaded into the DDR controller configuration registers from the eNVM.

The Configurator has the following tabs for entering configuration data:

- General data, such as DDR mode, data width, clock frequency, Error Correction Code (ECC), fabric interface, and drive strength
- Memory initialization data, such as burst length, burst order, timing mode, and latency
- Memory timing data

To enter memory timing data, perform the following steps:

1. Select the DDR type: **DDR2**, **DDR3**, or **LPDDR**.
2. Define the **DDR memory settling time**. Refer to your external DDR memory specifications to set the appropriate memory setting time. The DDR memory might fail to initialize correctly if the memory settling time is not set properly.
3. Either import the DDR register configuration data or set your DDR Memory Parameters. For more information, see the SmartFusion 2 and IGLOO 2 FPGA High-Speed DDR Interfaces User Guide.
4. See the specifications of your external DDR memory and configure the DDR controller to match the requirements of your external DDR memory. For information about DDR controller configuration registers, see the SmartFusion 2 and IGLOO 2 FPGA High-Speed DDR Interfaces User Guide.

The following figure shows an example of the configuration file syntax. The register names used in this file are the same as those described in the SmartFusion 2 and IGLOO 2 FPGA High-Speed DDR Interfaces User Guide.

Figure 2-3. Example of Configuration File Syntax

```
## PHY_16_DDR2_NO_ECC_BL8_INTER

ddrc_dyn_soft_reset_CR          0x00 ;
ddrc_dyn_refresh_1_CR           0x27DE ;
ddrc_dyn_refresh_2_CR           0x030F ;
ddrc_dyn_powerdown_CR           0x02 ;
ddrc_dyn_debug_CR               0x00 ;
ddrc_ecc_data_mask_CR           0x0000 ;
ddrc_addr_map_col_1_CR          0x3333 ;
ddrc_addr_map_col_3_CR          0x3300 ;
ddrc_init_1_CR                  0x0001 ;
ddrc_cke_rstn_cycles_CR1        0x0100 ;
ddrc_cke_rstn_cycles_CR2        0x0008 ;
ddrc_init_emr2_CR               0x0000 ;
ddrc_init_emr3_CR               0x0000 ;
ddrc_dram_bank_act_timing_CR    0x1947;
ddrc_odt_param_1_CR             0x0010 ;
ddrc_odt_param_2_CR             0x0000 ;
ddrc_debug_CR                   0x3300 ;
ddrc_mode_reg_rd_wr_CR          0x0000 ;
ddrc_mode_reg_data_CR           0x0000 ;
ddrc_pwr_save_2_CR              0x0000 ;
ddrc_hpr_queue_param_CR1        0x80F8 ;
ddrc_hpr_queue_param_CR2        0x0007 ;
ddrc_lpr_queue_param_CR1        0x80F8 ;
ddrc_lpr_queue_param_CR2        0x0007 ;
ddrc_wr_queue_param_CR          0x0200 ;
ddrc_dfi_min_ctrlupd_timing_CR  0x0003 ;
ddrc_dfi_max_ctrlupd_timing_CR  0x0040 ;
ddrc_dfi_wr_lvl_control_CR1     0x0000 ;
ddrc_dfi_wr_lvl_control_CR2     0x0000 ;
ddrc_dfi_rd_lvl_control_CR1     0x0000 ;
ddrc_dfi_rd_lvl_control_CR2     0x0000 ;
ddrc_dfi_ctrlupd_time_interval_CR  0x0309 ;
ddrc_perf_param_3_CR            0x0000 ;
ddrc_ecc_int_clr_reg            0x0000 ;
```

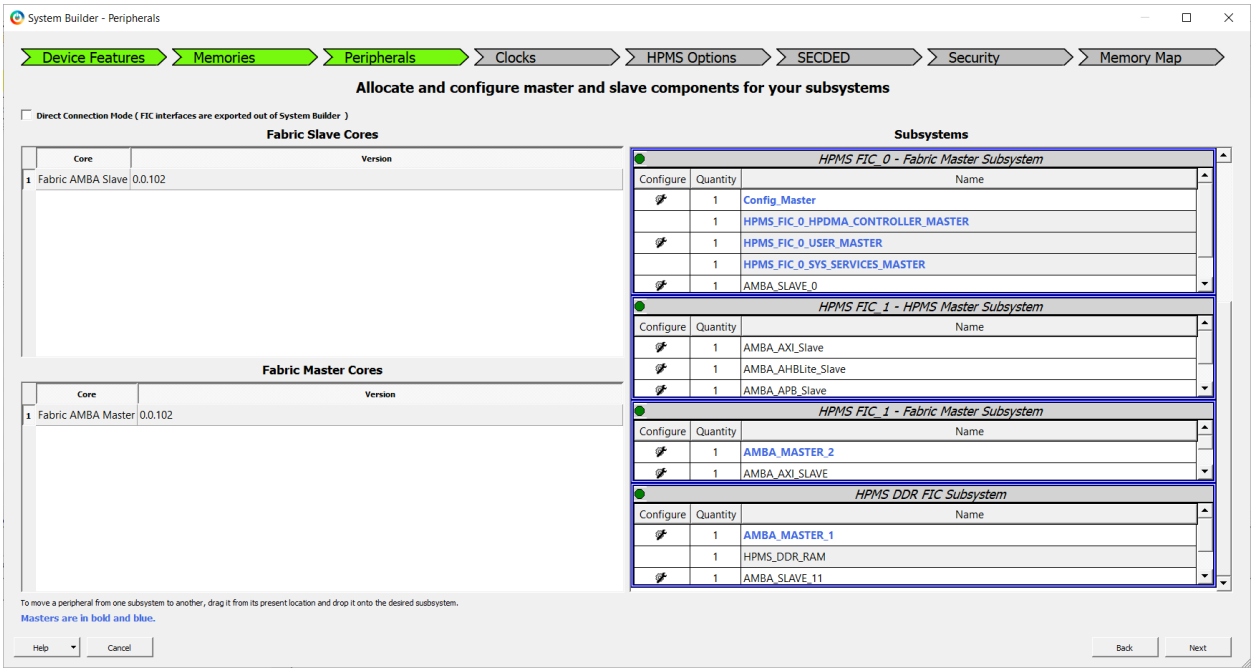
2.3 System Builder Peripherals Page (Ask a Question)

In the System Builder Peripherals page, a separate subsystem is created for each DDR controller (Fabric DDR Subsystem for FDDR and HPMS DDR FIC Subsystem for MDDR). You can add a Fabric AMBA® Master (configured as AXI/AHBLite) core to each of these subsystems to enable fabric master access to the DDR controllers. Upon generation, System Builder automatically:

- Instantiates bus cores, depending on the type of AMBA Master added
- Exposes the master BIF of the bus core and the clock and reset pins of the corresponding subsystems (FDDR/MDDR) under appropriate pin groups

Connect the BIFs and clock/reset pins to the appropriate Fabric Master cores that you want to instantiate.

Figure 2-4. System Builder Peripherals Page



2.4 System Builder Clock Settings Page (Ask a Question)

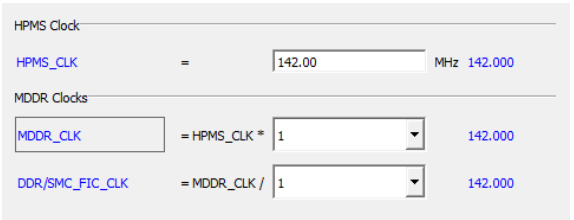
In the System Builder Clock Settings page, you specify the clock frequencies related to each DDR (MDDR and/or FDDR) subsystem.

For MDDR, specify the settings as listed in the following table.

Table 2-1. MDDR Settings

Setting	Description
MDDR_CLK	This clock determines the operating frequency of the DDR controller. It should match the clock frequency at which you want your external DDR memory to run. The MDDR_CLK must be less than 333 MHz. The MDDR_CLK is defined as a multiple of the HPMS_CLK.
DDR/SMC_FIC_CLK	If you chose to access the MDDR from the FPGA fabric, specify the DDR/SMC_FIC_CLK. This clock frequency is defined as ratio of the MDDR_CLK and should match the frequency, at which the FPGA fabric subsystem that accesses the MDDR is running.

Figure 2-5. HPMS Main Clock—MDDR Clocks

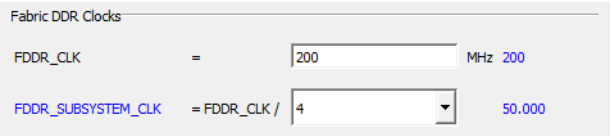


For FDDR, specify the settings as listed in the following table.

Table 2-2. FDDR Settings

Setting	Description
FDDR CLK	Determines the operating frequency of the DDR controller. This value should match the clock frequency at which you want your external DDR memory to run. The FDDR_CLK must range from 20 MHz and 333 MHz.
FDDR SUBSYSTEM CLK	The clock frequency is determined by the FDDR_CLK ratio and should be synchronized with the frequency of the FPGA fabric subsystem that accesses the FDDR. It is important to note that the FDDR_SUBSYSTEM_CLK should not exceed 200 MHz.

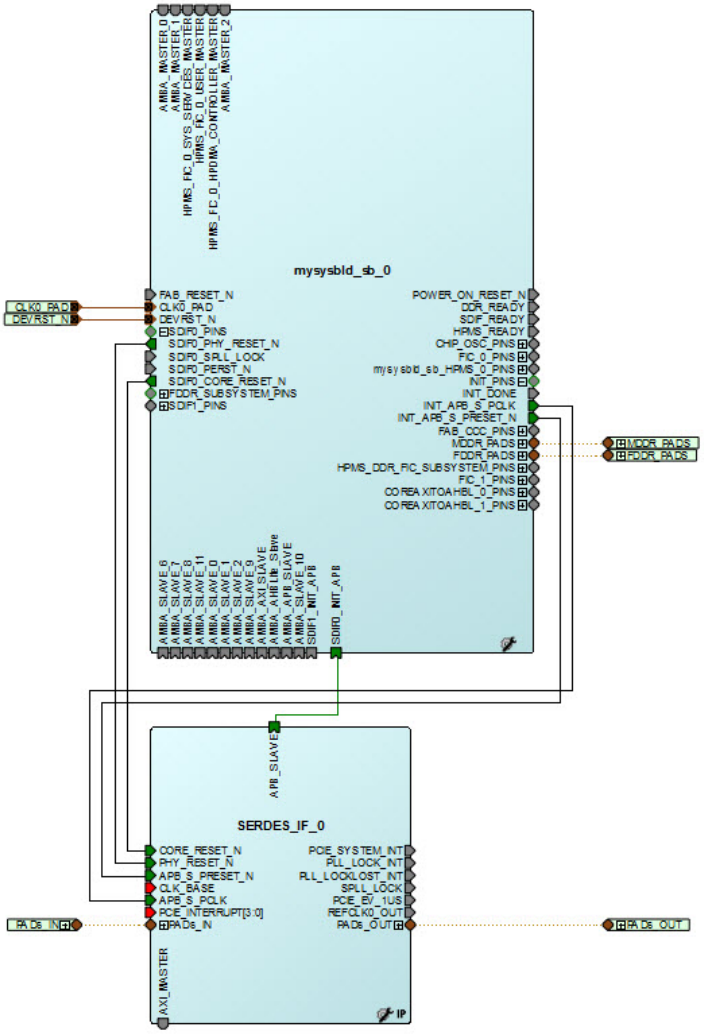
Figure 2-6. Fabric DDR Clocks—FDDR Clocks



2.5 SERDESIF Configuration (Ask a Question)

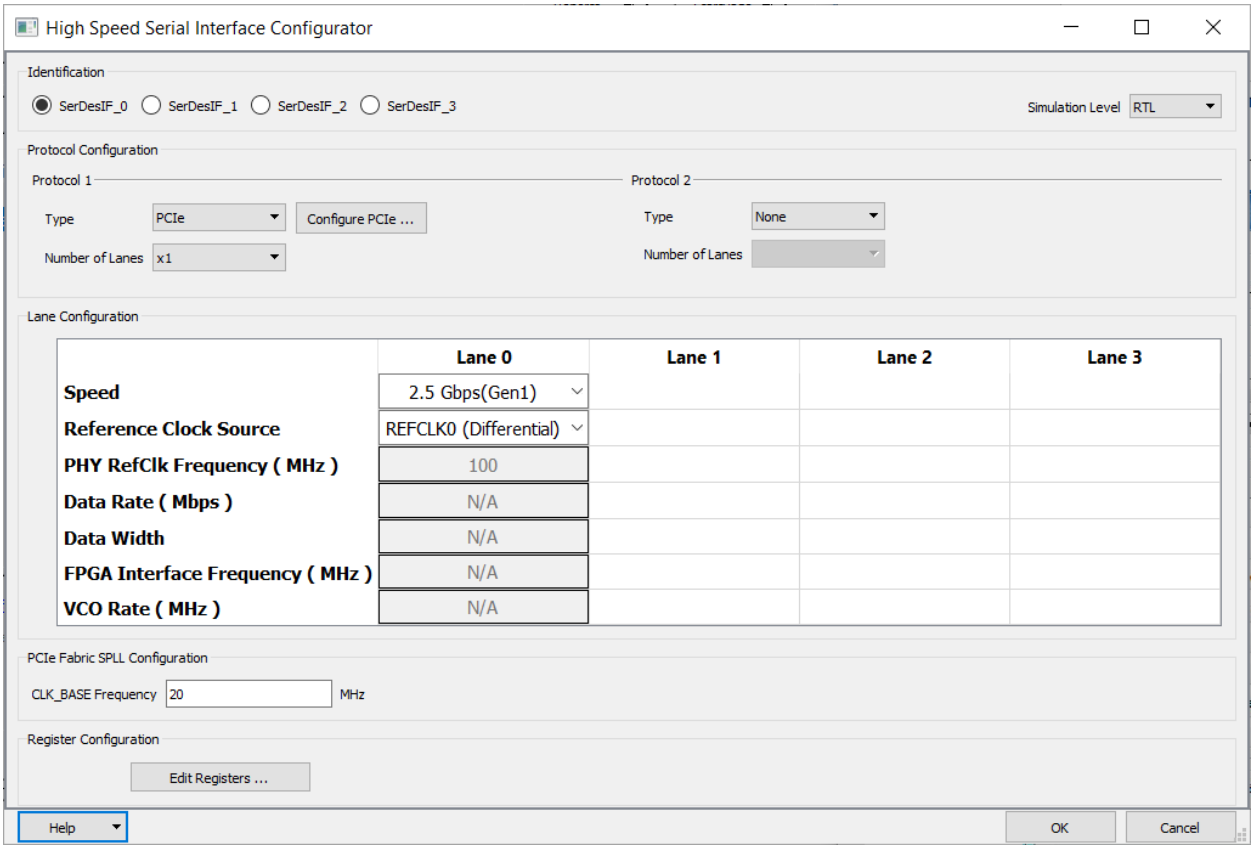
The SERDESIF blocks are not instantiated in the System Builder generated design. However, if SERDESIF locations are selected in the System Builder Device Features page while creating a design, then initialization logic for all the SERDESIF locations is built by System Builder. The corresponding initialization signals are available at the interface of the System Builder block, and can be connected to the SERDESIF blocks at the next level of hierarchy, as shown in the following figure.

Figure 2-7. SERDESIF Peripheral Initialization Connectivity



Similar to the DDR configuration registers, each SERDESIF block also has configuration registers that must be loaded at runtime. You can either import these register values or configure the UI parameters in the High Speed Serial Interface Configurator (see the following figure) to enter your PCIe or EPCS parameters. The register values are automatically computed for you. For more information, see the SmartFusion 2 and IGLOO 2 FPGA High-Speed Serial Interface User Guide.

Figure 2-8. High Speed Serial Interface Configurator



2.6 Continuing with the Design Flow (Ask a Question)

After you integrate your user logic with the System Builder block and SERDESIF block, generate your top-level SmartDesign. This generates all HDL files required to implement and simulate your design. You can then proceed with the rest of the design flow.

3. Initializing DDR and SERDES_IF Blocks (Ask a Question)

This section describes how the IGLOO 2 System Builder builds an initialization solution for the DDR and SERDESIF blocks. This section is designed to help you understand what the System Builder tool generates for you. This section describes how System Builder instantiates and connects the Fabric Cores required to transfer the configuration data to the DDR controllers and SERDESIF configuration registers.

To understand the design built by System Builder, right-click the System Builder component (**<design_name>_sb**) in the Libero Design Hierarchy window and select **Open as SmartDesign**. The System Builder design opens in regular SmartDesign window, showing all cores that System Builder instantiated, configured, and connected together. For more information about opening a System Builder block as a SmartDesign, see the topic *Opening a System Builder Block as SmartDesign* in the IGLOO2 System Builder User's Guide

If you open the System Builder block as a SmartDesign, the following sections will help you understand the initialization subsystem logic built by System Builder.

3.1 Creating the FPGA Design Initialization Subsystem (Ask a Question)

To initialize the DDR and SERDESIF blocks, System Builder creates the initialization subsystem in the FPGA fabric. When the FPGA fabric initialization subsystem resets, it loads peripheral configuration data from the eNVM into the DDR and SERDESIF configuration registers. It also manages the reset sequences required for these blocks to become operational and indicates when they are ready to communicate with the rest of your design.

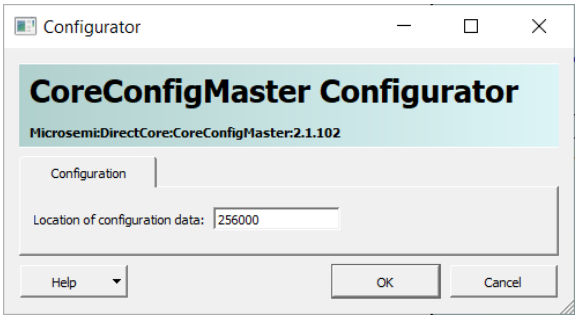
To create the initialization subsystem, System Builder:

- Instantiates and configures the CoreConfigMaster core.
- Configures `FIC_2` inside the HPMS.
- Instantiates and configures the `CoreConfigP` and `CoreResetP` cores.
- Instantiates and configures the Clock Conditioning Circuitry (CCC) block.
- Instantiates and configures the on-chip 25/50 MHz RC oscillator.
- Instantiates the System Reset (`SYSRESET`) macro.
- Connects these components to each peripheral's configuration interfaces, clocks, resets, and PLL lock ports.

3.1.1 CoreConfigMaster Configuration (Ask a Question)

System Builder instantiates and configures the CoreConfigMaster soft IP core into the `<design_name>_sb` component. System Builder configures the CoreConfigMaster to set the location of the configuration data in the HPMS eNVM based on the device being used. From this location in the eNVM, CoreConfigMaster fetches the peripheral configuration data and loads the configuration registers of the DDR and SERDESIF blocks (see the following figure).

Figure 3-1. CoreConfigMaster Example



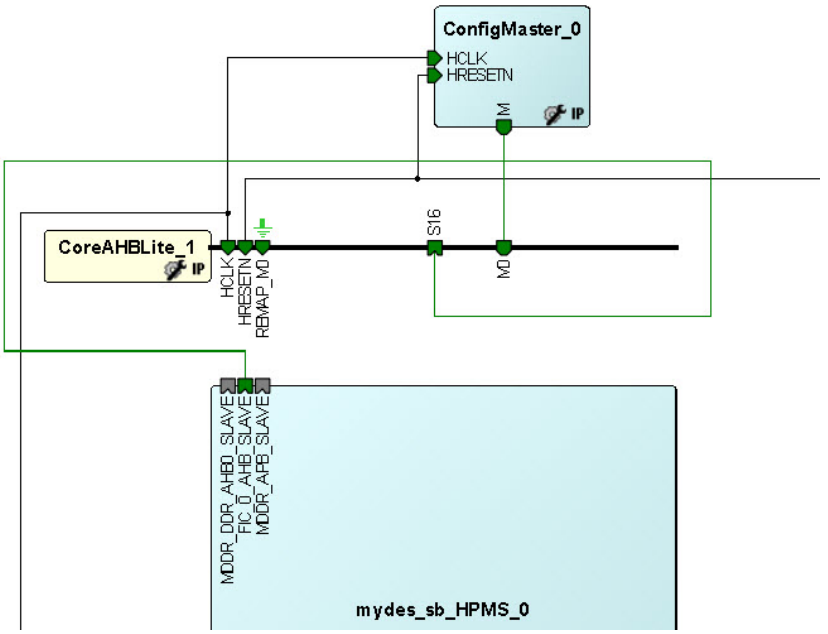
For the CoreConfigMaster to be able to fetch configuration data from the HPMS eNVM, System Builder establishes a bus-based subsystem (AMBA AHBLite) to provide access into the HPMS eNVM space. To establish this subsystem, System Builder:

1. Configures the `<design_name>_sb_HPMS` block to enable the Fabric Interface Controller `FIC_0`.
2. Configures the `FIC_0` to be of type AHBLite, and to use the Slave Interface that exposes the `FIC_0_AHB_SLAVE` bus interface on the HPMS block.
3. Instantiates and configures CoreAHBLite soft IP bus core, and configures its address space so that the CoreConfigMaster mastering the bus core can access the eNVM space in the HPMS.
4. Connects the master bus interface of the CoreConfigMaster to one of the CoreAHBLite mirrored master bus interfaces.
5. Connects the S16 slave bus interface of CoreAHBLite to the `FIC_0_AHB_SLAVE` slave bus interface of the HPMS block.

This data path via CoreAHBLite provides access to the HPMS eNVM space for the CoreConfigMaster, so that it can fetch the configuration data of the DDR controllers and the SERDESIF blocks.

The configuration data that the CoreConfigMaster fetches from the HPMS eNVM is directed onto the `HPMS FIC_2` interface via the AHB Bus Matrix (the SWITCH) inside the HPMS block. The HPMS `FIC_2` interface, which is an APB3 master bus interface, directs this configuration data to the DDR controllers and the SERDESIF blocks via fabric cores such as CoreConfigP and CoreResetP (see the following figure).

Figure 3-2. CoreAHBLite Bus Subsystem



3.1.2 HPMS FIC_2 APB Configuration (Ask a Question)

Fabric Interface Controller 2 (FIC_2) in the HPMS is an APB Master of the configuration path to the FDDR and SerDes.

To see the HPMS FIC_2 configuration:

1. Double-click the `<design_name>_sb_HPMS` block in the SmartDesign window to open the HPMS Configurator.
2. Open the `FIC_2` configurator dialog from the HPMS configurator.
3. Depending on your system, one or both of the following options must be selected by System Builder to initialize:
   - HPMS DDR
   - Fabric DDR and/or SerDes Blocks
4. Observe that the `FIC_2` ports (`FIC_2_APB_MASTER`, `FIC_2_APB_M_PCLK`, and `FIC_2_APB_M_PRESET_N`) are exposed at the HPMS interface and are connected to the CoreConfigP and CoreResetP cores by System Builder.

3.1.3 CoreConfigP Configuration (Ask a Question)

System Builder instantiates and configures the CoreConfigP soft IP core into the `<design_name>_sb` component. To see how System Builder configured the CoreConfigP core, double-click the core to open its configurator. Observe that:

- System Builder selects the appropriate check boxes to show which peripherals you use in your design based on your selection in the System Builder Device Features page.
- For each SerDes block in your design, System Builder selects the appropriate check boxes to indicate whether PCIe is used or not.

Note: If you use an 090 or 060 device (M2GL090 or M2GL060) and your design uses SERDESIF, System Builder does not select the **SDIF0 used for PCIe** check box. If you use a non-M2GL090/M2GL060 device and use one or more SERDESIF blocks, System Builder selects the **SDIF<0-3> used for PCIe** check box under the appropriate SERDESIF section.

For information about the options in this configurator, see the *CoreConfigP Handbook*.

Figure 3-3. CoreConfigP Example

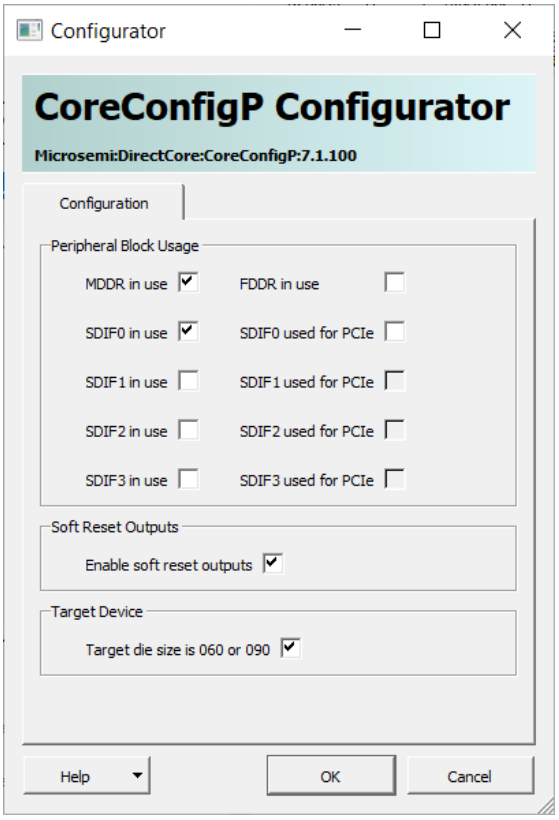
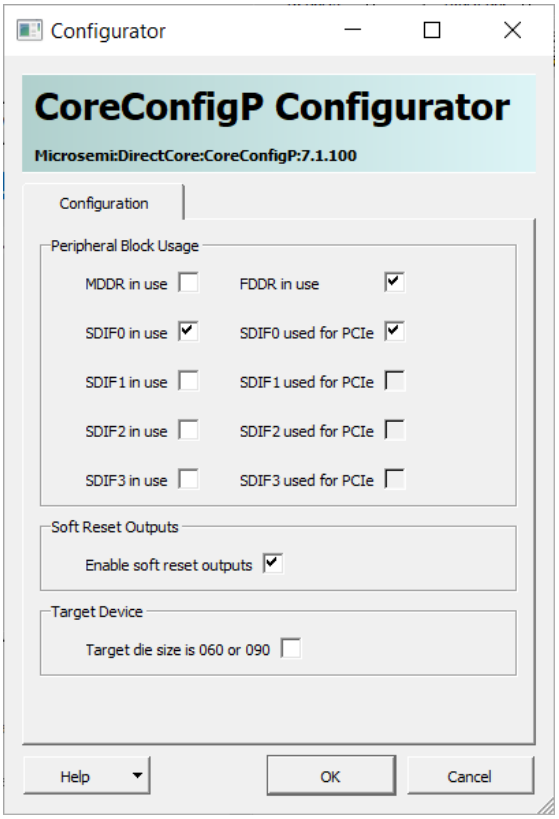


Figure 3-4. CoreConfigP Configurator for M2GL090/060 Devices



3.1.4 CoreResetP Configuration (Ask a Question)

System Builder instantiates and configures the CoreResetP soft IP core into the `<design_name>_sb` component. To see how System Builder configured the CoreResetP core, double-click the core to open its configurator. Observe that:

- System Builder specifies the external reset behavior (`EXT_RESET_OUT` asserted).
- System Builder specifies the Device Voltage. The selected value matches the voltage you selected in the Libero **Project Settings** dialog box.
- System Builder selects the appropriate check boxes to show which peripherals you use in your design based on your selection in the System Builder Device Features page.
- System Builder specifies the external DDR memory setting time. This is the maximum value for all DDR memories used in your application (MDDR and FDDR). Refer to the DDR Memory Datasheet from external vendor to configure this parameter.
  Note: This is an important parameter to guarantee a working simulation and a working system on silicon. Incorrect value for the settling time might result in simulation errors.

Tip: 200 µs is a good default value for DDR2 and DDR3 memories running at 200 MHz.

- For each SerDes block in your design, System Builder selects the appropriate boxes to indicate whether PCIe is used, support for PCIe Hot Reset is required, and whether support for PCIe L2/P2 is required.

Note: If you use an 090 or 060 device (M2GL090 or M2GL060) and your design uses SERDESIF, System Builder does not select the **Used for PCIe**, **Include PCIe HotReset support**, and **Include PCIe L2/P2 support** check boxes. If you use a non-M2GL090/M2GL060 device and use one or more SERDESIF blocks, System Builder selects all four check boxes under the appropriate SERDESIF section.

For information about the options in this configurator, see the *CoreResetP User Guide* in the Libero Catalog.

Figure 3-5. CoreResetP Configurator Example

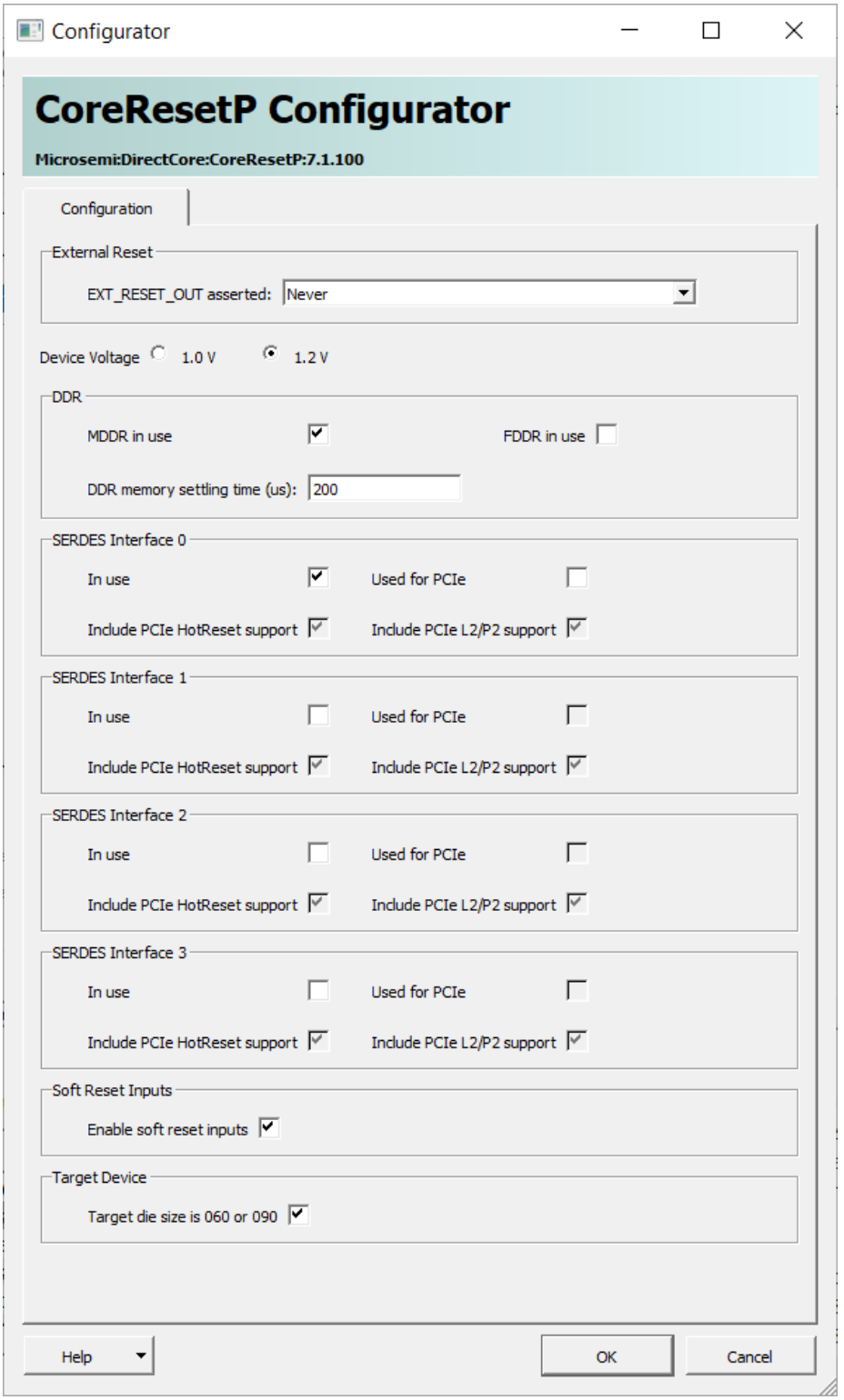
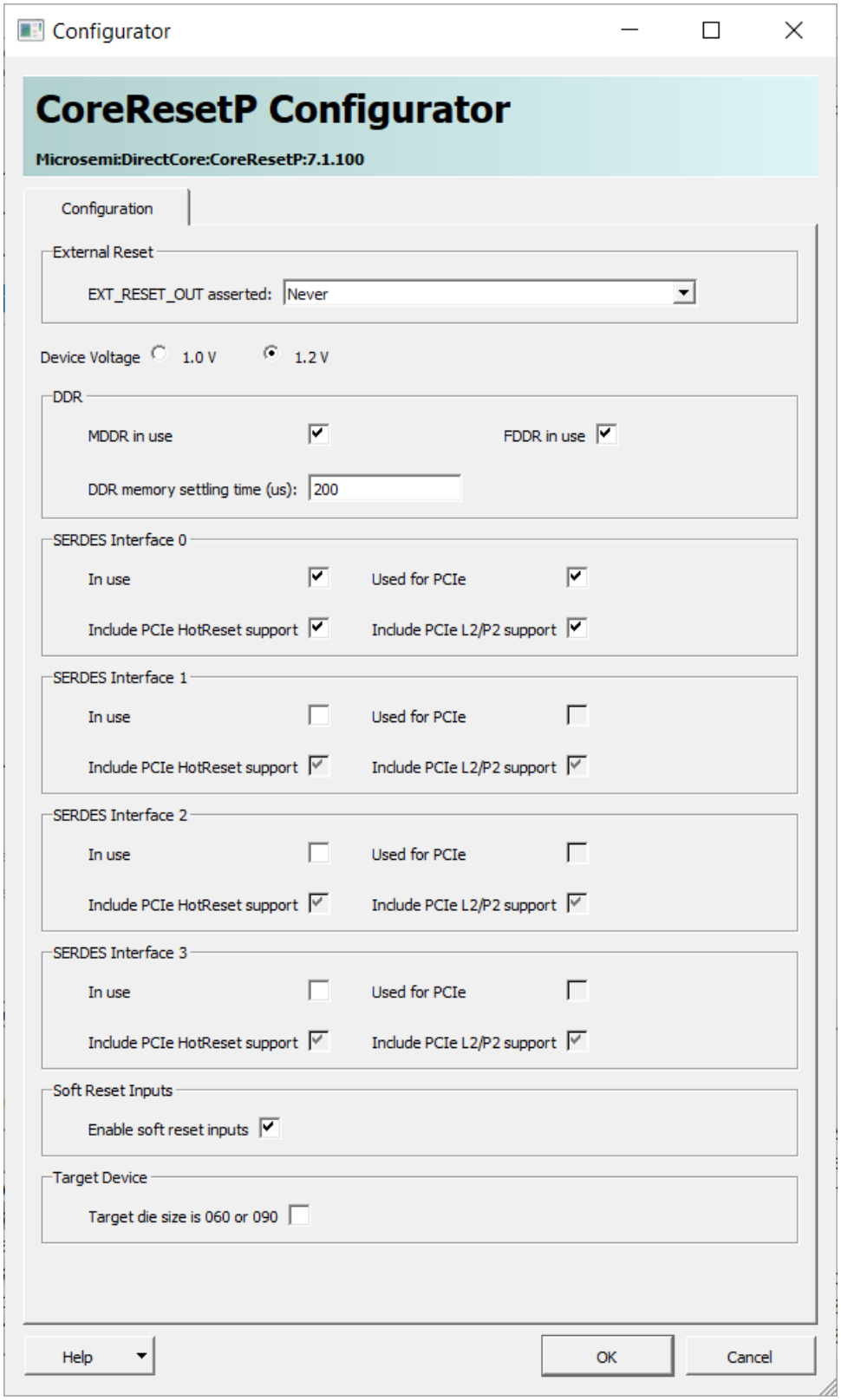


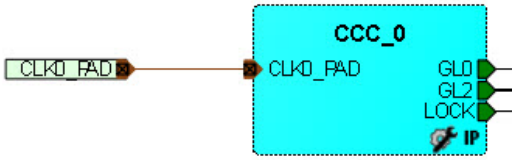
Figure 3-6. CoreResetP Configurator for M2GL060/090 Devices (SERDES_0 Used)



3.1.5 Clock Conditioning Circuitry (CCC) Instantiation (Ask a Question)

Based on the clock configuration set in the System Builder Clock Settings page, System Builder instantiates and configures the CCC block in the `<design_name>_sb` component automatically to expose clock globals (GL<0-3>) that match certain subsystem clock frequencies and other required clocks. System Builder promotes and exposes these clock globals (GL<0-3>) and the `LOCK` output signal of the CCC block to the top level under different pin groups on the System Builder block interface, depending on the system.

Figure 3-7. CCC Instantiation

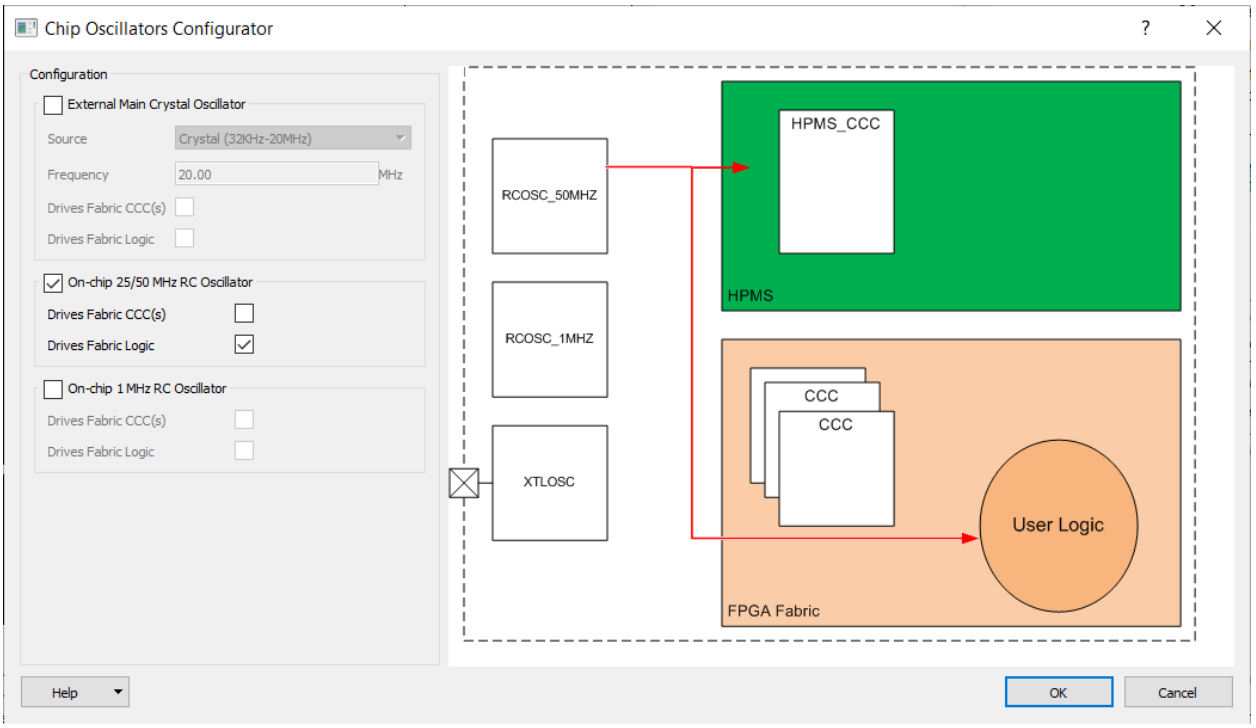


3.1.6 50 MHz Oscillator Instantiation (Ask a Question)

CoreConfigP and CoreResetP are clocked by the on-chip 25/50 MHz RC oscillator. System Builder instantiates a 25/50 MHz oscillator and connects it to these cores. The following steps describe how System Builder instantiates and configures the chip oscillators core:

1. Instantiates the Chip Oscillators core into the `<design_name>_sb` component. Typically, this is where the HPMS is instantiated. This core can be found in the Libero Catalog under Clock & Management.
2. Configures this core such that the oscillator drives the FPGA fabric, as shown in the following figure.
3. Connects the `RCOSC_25_50_MHz_O2F` output of the oscillator to the `RCOSC_25_50` MHz input of CoreResetP.

Figure 3-8. Chip Oscillators Configurator



3.1.7 System Reset (SYSRESET) Instantiation (Ask a Question)

The SYSRESET macro provides device-level reset functionality to your design. The POWER_ON_RESET_N output signal is asserted or de-asserted whenever the chip is powered up or the external pin DEVRST_N is asserted or de-asserted (see the following figure).

System Builder instantiates the SYSRESET macro into the <design_name>_sb component (typically, where the HPMS is instantiated). This macro can be found in the Libero Catalog under Macro Library. No configuration of this macro is necessary. System Builder connects the POWER_ON_RESET_N output of the SYSRESET macro to the POWER_ON_RESET_N input of CoreResetP and promotes it to top level, so that you can use it to reset any fabric logic upon power-up.

Figure 3-9. SYSRESET Macro

3.1.8 Overall Connectivity (Ask a Question)

After instantiating and configuring the HPMS, FDDR, OSC, SYSRESET, CoreConfigP, and CoreResetP cores in your design, System Builder connects them to the peripheral initialization subsystem. For SERDESIF blocks that you instantiate manually in your SmartDesign, manually connect the pins corresponding to the peripheral initialization subsystem (bus interface, clock, reset, and other pins) that System Builder exposes to the SERDESIF blocks. To simplify the connectivity description in this guide, the APB3-compliant configuration data path connectivity associated with the CoreConfigP and the CoreResetP related connections is described.

3.1.9 Configuration Data Path Connectivity (Ask a Question)

The following table and figure show how System Builder connects the CoreConfigP to the HPMS FIC_2 signals and the peripherals' APB3-compliant configuration interfaces. System Builder connects the signals corresponding to the FDDR and the MDDR; however, you must manually connect the signals corresponding to the SERDESIF blocks after the SERDESIF blocks are instantiated in the SmartDesign.

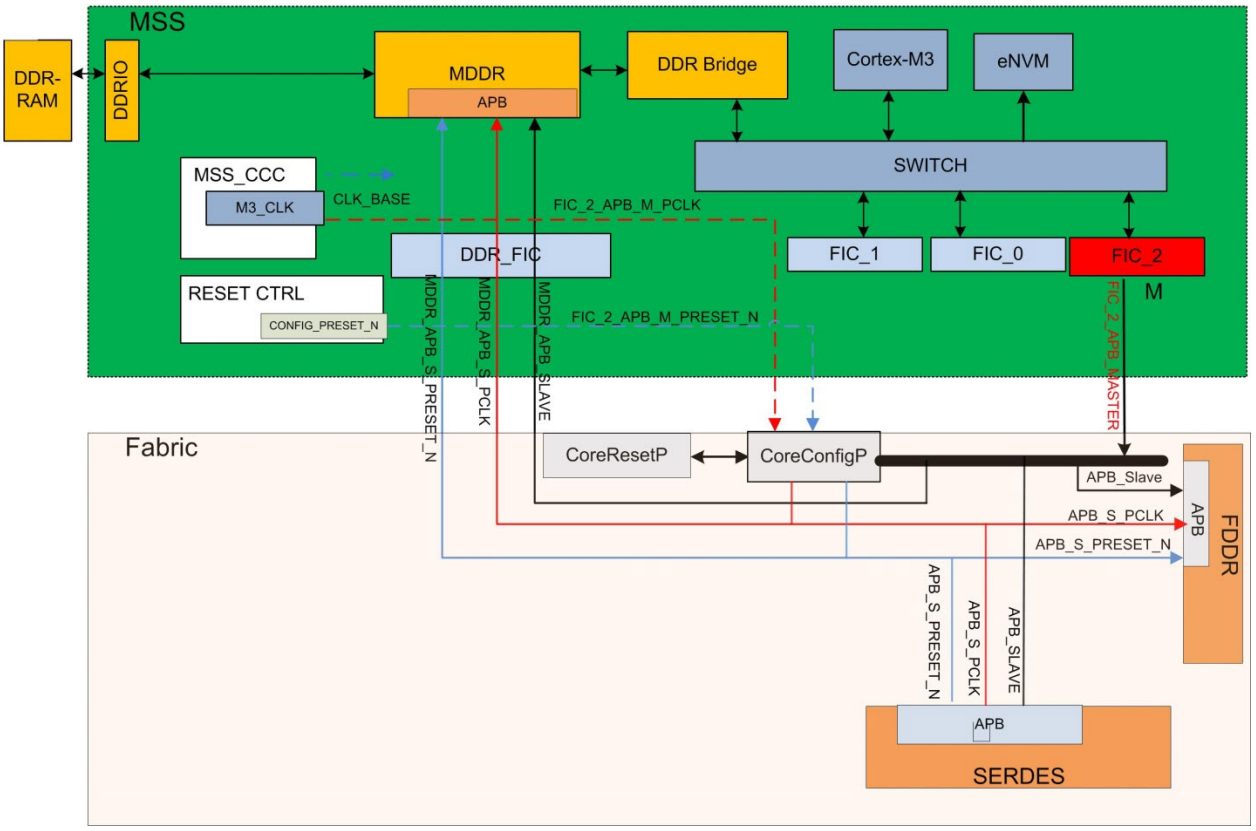
Table 3-1. Configuration Data Path Port/BIF Connections

FROM Port or Bus Interface (BIF)/ Component	TO Port or Bus Interface (BIF)/Component		
	SERDESIF Pins	FDDR Pins	MDDR Pins
APB_S_PRESET_N/ CoreConfigP	APB_S_PRESET_N/ SDIF<0/1/2/3>	APB_S_PRESET_N/FDDR	MDDR_APB_S_PRES ET_N/ HPMS
APB_S_PCLK/ CoreConfigP	APB_S_PCLK/SDIF<0/1/2/3>	APB_S_PCLK/FDDR	MDDR_APB_S_PCLK/HPMS
MDDR_APBmslave/ CoreConfigP	—	—	MDDR_APB_SLAVE (BIF)/HPMS
SDIF<0/1/2/ 3>_APBmslave/ CoreConfigP	APB_SLAVE (BIF)/SDIF<0/1/2/3>	—	—
FDDR_APBmslave/ CoreConfigP	—	APB_SLAVE (BIF)/FDDR	—

.....continued

FROM Port or Bus Interface (BIF)/ Component	TO Port or Bus Interface (BIF)/Component		
	SERDESIF Pins	FDDR Pins	MDDR Pins
FIC_2_APBmmaster /CoreConfigP	—	—	FIC_2_APB_MASTER/HPMS

Figure 3-10. FIC_2 APB3 Subsystem Connectivity

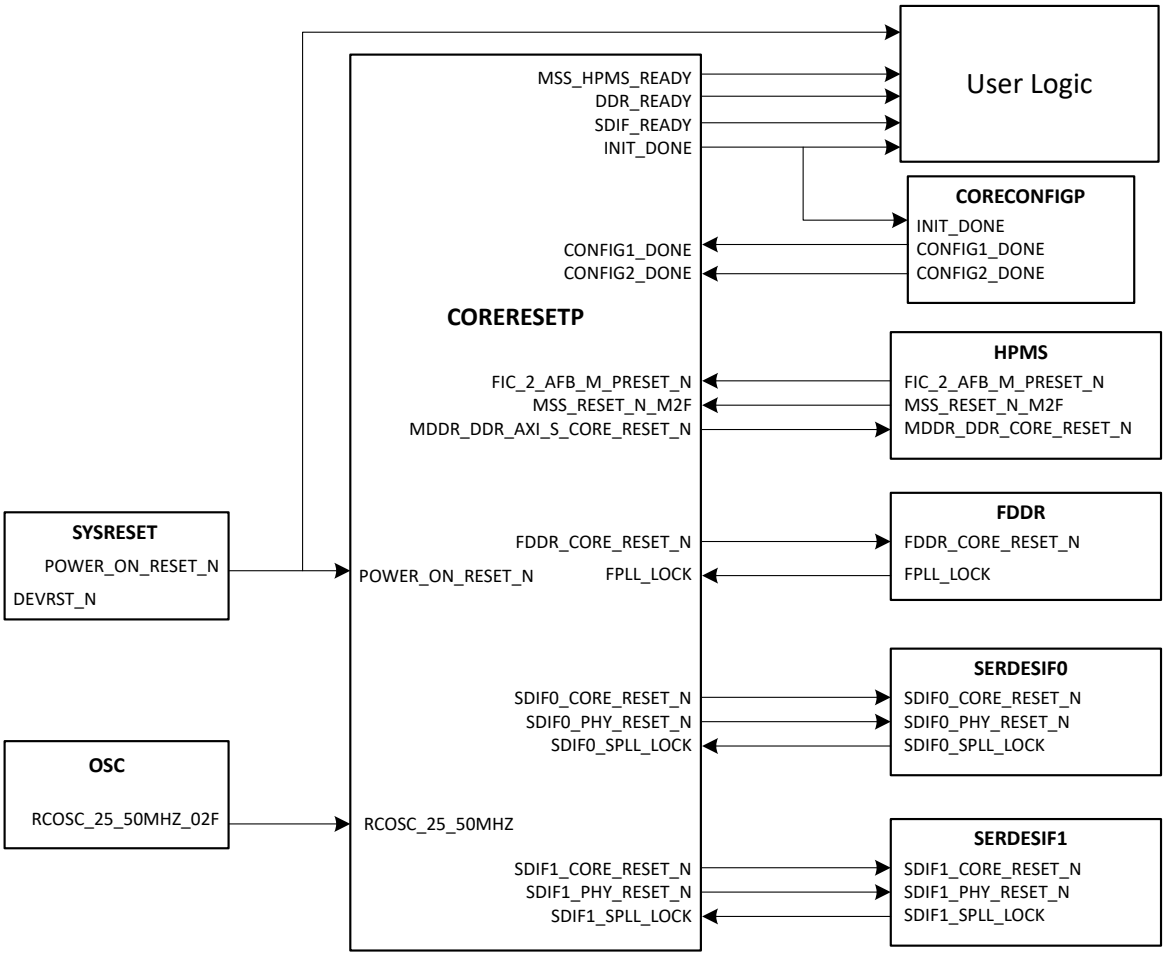


3.1.10 Clocks and Resets Connectivity (Ask a Question)

The following figure summarizes clocks and resets connectivity. This figure shows how:

- System Builder connects the CoreResetP core to the external reset sources and the peripherals' core reset signals.
- System Builder connects the CoreResetP to the peripherals' clock synchronization status signals (PLL lock signals).
- The CoreConfigP and CoreResetP are connected.

Figure 3-11. CoreResetP Subsystem Connectivity



4. Configuration and Simulation File-set (Ask a Question)

After configuring different registers of the DDR controllers (FDDR/MDDR) in the System Builder Memories page and generating the design in System Builder, the configuration files with extension .reg are created in the appropriate component folders in the <project dir>/component/work/ directory.

For SERDES_IF, after configuring different registers in the SERDES_IF block and generating the SmartDesign component in which the SERDES_IF block is instantiated, the configuration files with the extension .reg that correspond to the SERDES_IF_n (where, n = 0, 1, 2, 3) blocks are created in the appropriate component folders in the <project dir>/component/work/ directory.

In addition, simulation files corresponding to SerDes PCIe BFM simulations are created in the <project_dir>/simulation directory.

The following sections describe the files, where they are generated, their contents, and which tools use them.

4.1 MDDR_init.reg File (Ask a Question)

The MDDR_init.reg file is generated to the <project_dir>/component/work/<design_name>_sb_HPMS directory after generating the System Builder component.

This file contains the MDDR register configuration data.

This file is used by the Simulation, Programming, and Power tools.

4.2 FDDR_init.reg File (Ask a Question)

The FDDR_init.reg file is generated to the <project_dir>/component/work/<design_name>_sb/FABDDR_0 directory after generating the System Builder component containing the FDDR block.

This file contains the FDDR register configuration data.

This file is used by the Simulation, Programming, and Power tools.

4.3 SERDESIF_<0/1/2/3>_init.reg File (Ask a Question)

The SERDESIF_<0/1/2/3>_init.reg file is generated to the <project_dir>/component/work/<SmartDesign_Comp>/SERDESIF_n directory (where n = 0, 1, 2, or 3), after generating the SD component containing the SERDESIF_n block.

This file contains the SerDes register configuration data.

This file is used by the Simulation, Programming, and Power tools.

4.4 SERDESIF_<0/1/2/3>_user.bfm File (Ask a Question)

The SERDESIF_<0/1/2/3>_user.bfm file is generated to the <project_dir>/simulation directory.

This file contains the user commands. Edit this file to enter your BFM commands. Use this file if you configured SERDESIF_n (where n = 0, 1, 2, or 3) block in BFM PCIe simulation mode and as an AXI/AHBLite master. If you configured the SERDESIF_n block in RTL simulation mode, you will not need this file.

Each time you start the simulation, these simulation files are recreated to the <project dir>/simulation directory with updated contents.

4.5 ENVM_init.mem File (Ask a Question)

The ENVM_init.mem file is generated to the <project_dir>/simulation directory.

This file contains the register configuration data in binary format corresponding to all the DDR controllers (FDDR/MDDR) and the SERDES_IF blocks used in your design.

The contents of this file correspond to the contents fetched from the HPMS eNVM and are loaded to the peripherals' configuration registers during the simulation runtime.

When you run a programming tool (Generate Bitstream or Export Bitstream) in the design flow, this file is created to the project directory.

4.6 root_comp_name>.efc File (Ask a Question)

The root_comp_name>.efc file is generated to the <project dir>/designer/<root_comp_name> directory.

This file contains the register configuration data in hexadecimal format corresponding to all the DDR controllers (FDDR/MDDR) and the SERDES_IF blocks used in your design.

The contents of this file are loaded to the HPMS eNVM on-board when the device is programmed.

Upon reset, these contents load into the eNVM, where they are fetched and loaded to the peripherals' configuration registers.

5. 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
A	10/2023	The following is the list of changes in revision A of the document: • The document was migrated to the Microchip template. • The document number was updated from 50200384 to DS50003599. • Information about configuration of RCOSC is changed from fabric to CCC drive. See 3.1.6. 50 MHz Oscillator Instantiation.

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

The Microchip Website

Microchip provides online support via our website at www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: www.microchip.com/support

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 product is 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.

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.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, TimeCesium, TimeHub, TimePictra, TimeProvider, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic

ISBN: 978-1-6683-3294-8

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office	**Australia - Sydney**	**India - Bangalore**	**Austria - Wels**
2355 West Chandler Blvd.	Tel: 61-2-9868-6733	Tel: 91-80-3090-4444	Tel: 43-7242-2244-39
Chandler, AZ 85224-6199	**China - Beijing**	**India - New Delhi**	Fax: 43-7242-2244-393
Tel: 480-792-7200	Tel: 86-10-8569-7000	Tel: 91-11-4160-8631	**Denmark - Copenhagen**
Fax: 480-792-7277	**China - Chengdu**	**India - Pune**	Tel: 45-4485-5910
Technical Support:	Tel: 86-28-8665-5511	Tel: 91-20-4121-0141	Fax: 45-4485-2829
www.microchip.com/support	**China - Chongqing**	**Japan - Osaka**	**Finland - Espoo**
Web Address:	Tel: 86-23-8980-9588	Tel: 81-6-6152-7160	Tel: 358-9-4520-820
www.microchip.com	**China - Dongguan**	**Japan - Tokyo**	**France - Paris**
Atlanta	Tel: 86-769-8702-9880	Tel: 81-3-6880- 3770	Tel: 33-1-69-53-63-20
Duluth, GA	**China - Guangzhou**	**Korea - Daegu**	Fax: 33-1-69-30-90-79
Tel: 678-957-9614	Tel: 86-20-8755-8029	Tel: 82-53-744-4301	**Germany - Garching**
Fax: 678-957-1455	**China - Hangzhou**	**Korea - Seoul**	Tel: 49-8931-9700
Austin, TX	Tel: 86-571-8792-8115	Tel: 82-2-554-7200	**Germany - Haan**
Tel: 512-257-3370	**China - Hong Kong SAR**	**Malaysia - Kuala Lumpur**	Tel: 49-2129-3766400
Boston	Tel: 852-2943-5100	Tel: 60-3-7651-7906	**Germany - Heilbronn**
Westborough, MA	**China - Nanjing**	**Malaysia - Penang**	Tel: 49-7131-72400
Tel: 774-760-0087	Tel: 86-25-8473-2460	Tel: 60-4-227-8870	**Germany - Karlsruhe**
Fax: 774-760-0088	**China - Qingdao**	**Philippines - Manila**	Tel: 49-721-625370
Chicago	Tel: 86-532-8502-7355	Tel: 63-2-634-9065	**Germany - Munich**
Itasca, IL	**China - Shanghai**	**Singapore**	Tel: 49-89-627-144-0
Tel: 630-285-0071	Tel: 86-21-3326-8000	Tel: 65-6334-8870	Fax: 49-89-627-144-44
Fax: 630-285-0075	**China - Shenyang**	**Taiwan - Hsin Chu**	**Germany - Rosenheim**
Dallas	Tel: 86-24-2334-2829	Tel: 886-3-577-8366	Tel: 49-8031-354-560
Addison, TX	**China - Shenzhen**	**Taiwan - Kaohsiung**	**Israel - Ra'anana**
Tel: 972-818-7423	Tel: 86-755-8864-2200	Tel: 886-7-213-7830	Tel: 972-9-744-7705
Fax: 972-818-2924	**China - Suzhou**	**Taiwan - Taipei**	**Italy - Milan**
Detroit	Tel: 86-186-6233-1526	Tel: 886-2-2508-8600	Tel: 39-0331-742611
Novi, MI	**China - Wuhan**	**Thailand - Bangkok**	Fax: 39-0331-466781
Tel: 248-848-4000	Tel: 86-27-5980-5300	Tel: 66-2-694-1351	**Italy - Padova**
Houston, TX	**China - Xian**	**Vietnam - Ho Chi Minh**	Tel: 39-049-7625286
Tel: 281-894-5983	Tel: 86-29-8833-7252	Tel: 84-28-5448-2100	**Netherlands - Drunen**
Indianapolis	**China - Xiamen**		Tel: 31-416-690399
Noblesville, IN	Tel: 86-592-2388138		Fax: 31-416-690340
Tel: 317-773-8323	**China - Zhuhai**		**Norway - Trondheim**
Fax: 317-773-5453	Tel: 86-756-3210040		Tel: 47-72884388
Tel: 317-536-2380			**Poland - Warsaw**
Los Angeles			Tel: 48-22-3325737
Mission Viejo, CA			**Romania - Bucharest**
Tel: 949-462-9523			Tel: 40-21-407-87-50
Fax: 949-462-9608			**Spain - Madrid**
Tel: 951-273-7800			Tel: 34-91-708-08-90
Raleigh, NC			Fax: 34-91-708-08-91
Tel: 919-844-7510			**Sweden - Gothenberg**
New York, NY			Tel: 46-31-704-60-40
Tel: 631-435-6000			**Sweden - Stockholm**
San Jose, CA			Tel: 46-8-5090-4654
Tel: 408-735-9110			**UK - Wokingham**
Tel: 408-436-4270			Tel: 44-118-921-5800
Canada - Toronto			Fax: 44-118-921-5820
Tel: 905-695-1980			
Fax: 905-695-2078			