SmartFusion2

System Builder User's Guide

Microsemi
Power Matters.™

Table of Contents

Introduction

System Builder is a graphical design wizard created to help you take advantage of the silicon features of the SmartFusion2 device to build a correct-by-design block. System Builder takes you through the following steps:

- Selecting the Device Features for your system
- Creating AMBA-based subsystems that connect to the SmartFusion2 Microcontroller Subsystem (MSS) Fabric Interface Controllers (FICs)
- Setting required configuration options for each selected feature
- Building a correct-by-design complete system

The System Builder wizard (Figure 1) creates your design based on high level design specifications that define your intended system. System Builder enables you to focus on your design needs instead of on the specific silicon requirements of a SmartFusion2 based design.

This simplifies the design creation process. The built-in design rule check feature prevents you from moving forward if there are any errors or conflicts.

System Builder uses your feature selections to instantiate, configure, and connect the necessary low level blocks to achieve your requirements. This automates what you would have had to do manually in SmartDesign.

The design block produced by System Builder follows all silicon design rules.

It builds the configuration and initialization circuitry, the clock tree, the AMBA bus structure, the reset circuitry and the interrupt circuitry your system needs. You can then connect the peripherals, such as SERDES blocks or external DDR memories, to the System Builder block using SmartDesign.

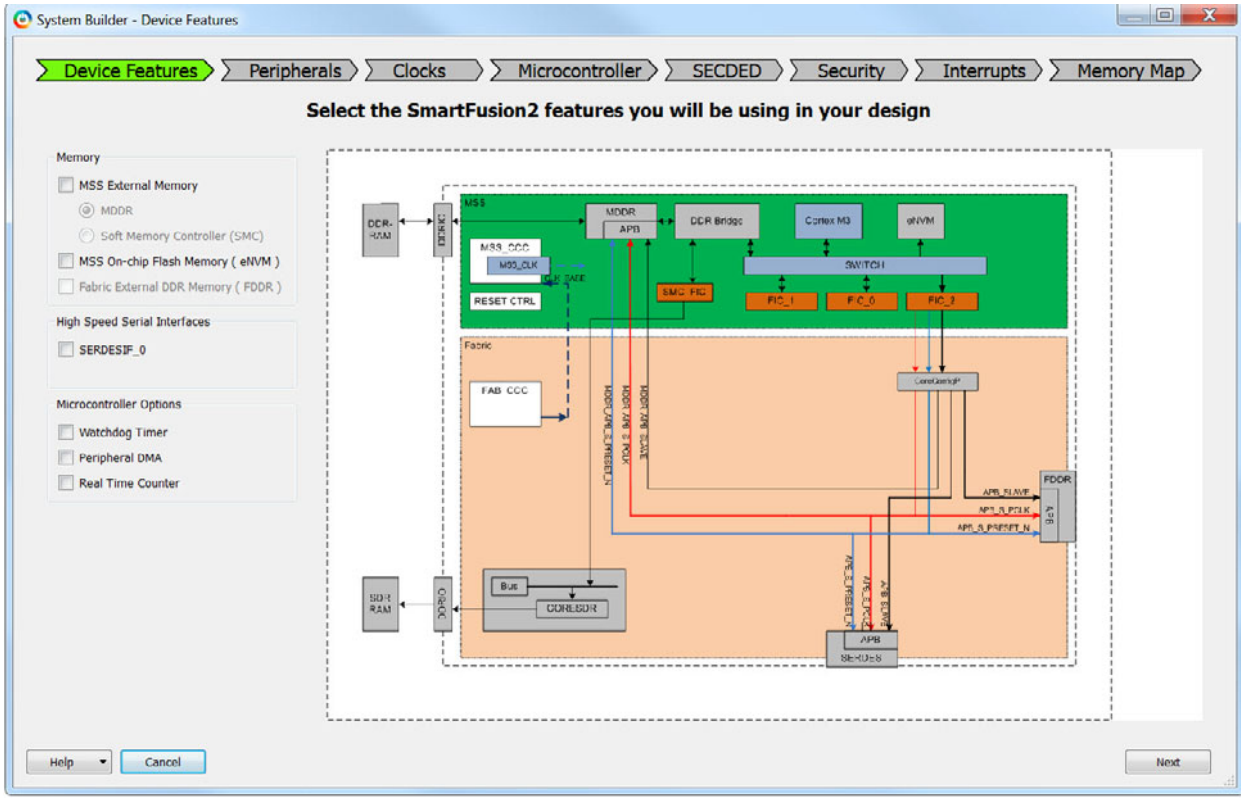

***Figure 1* • Selecting Device Features in System Builder**

1 – Accessing System Builder

You can access System Builder when you first create a project for SmartFusion2 in the Design Template page of the New Project Wizard (Figure 1-1).To create a System Builder component in your new project, select **Create a System Builder based design**.

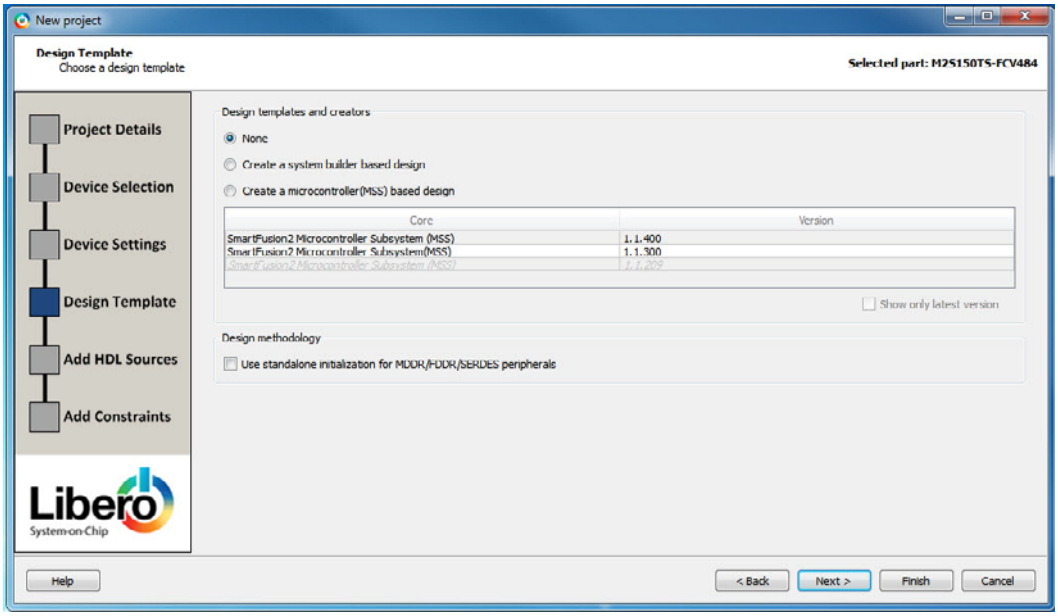

Figure 1-1 • **Design Template — Create a System Builder based design**

If you have already created your project, you can invoke System Builder from the Design Flow window (Figure 1-2).

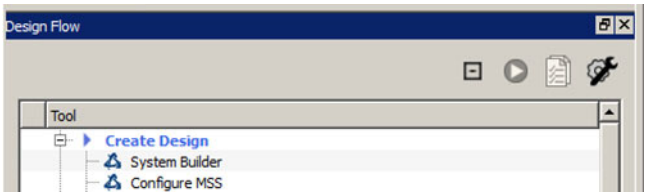

Figure 1-2 • **System Builder in the Design Flow Window**

For System Builder to function correctly, a set of required cores must be in your vault. These cores are required and used by System Builder to build the peripheral configuration/ initialization circuitry, the reset circuitry and the clock tree. These cores include but are not limited to:

- CoreResetP
- CoreConfigP
- Clock-Conditioning Circuitry (CCC)
- Oscillator (OSC)
- MSS Microcontroller
- Bus/Bridge Cores:

  - CoreAHBLite
  - CoreAPB3
  - CoreAXI
  - CoreAHBtoAPB
  - CoreAXItoAHBL
  - CoreAHBLtoAXI
- Power-on System Reset Core (SYSRESET_POR)

For details about the Configuration and Initialization of subsystems, refer to the SmartFusion2 DDR Controller and Serial High Speed Controller Initialization Methodology User Guide.

This set of required cores is downloaded automatically if you are connected to the Internet when System Builder is invoked. If these cores are not in your vault and cannot be downloaded from the core repositories, System Builder displays a message indicating that certain cores are not available in your current vault. You must either connect to the Internet to allow System Builder to download all the required cores or change your vault settings (Project menu > vault/repositories settings) to point to an existing vault that has these cores available.

2 – System Builder Configuration Pages

System Builder steps you through the following Configuration pages at the top of the System Builder Wizard for you to configure your design (Figure 2-1):

- Device Features
- Memories
- Peripherals
- Clocks
- Microcontroller
- SECDED
- Security
- Interrupts
- Memory Map

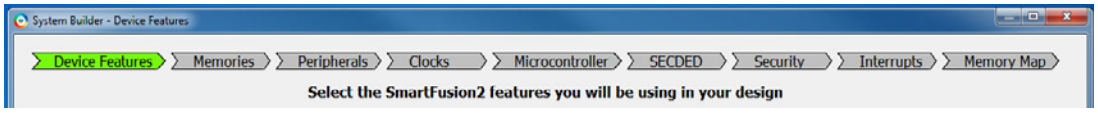

***Figure 2-1* • System Builder Configuration Pages**

Device Features Page

The Device Features page allows you to specify the Fabric and Microcontroller features you want for your system. If your plan to use MSS External Memory (DDR RAMs or SDRAMs), Fabric External Memory (DDR RAMs), the eNVM, or SERDES blocks in the fabric, the Device Features page is where you specify these features (Figure 2-2).

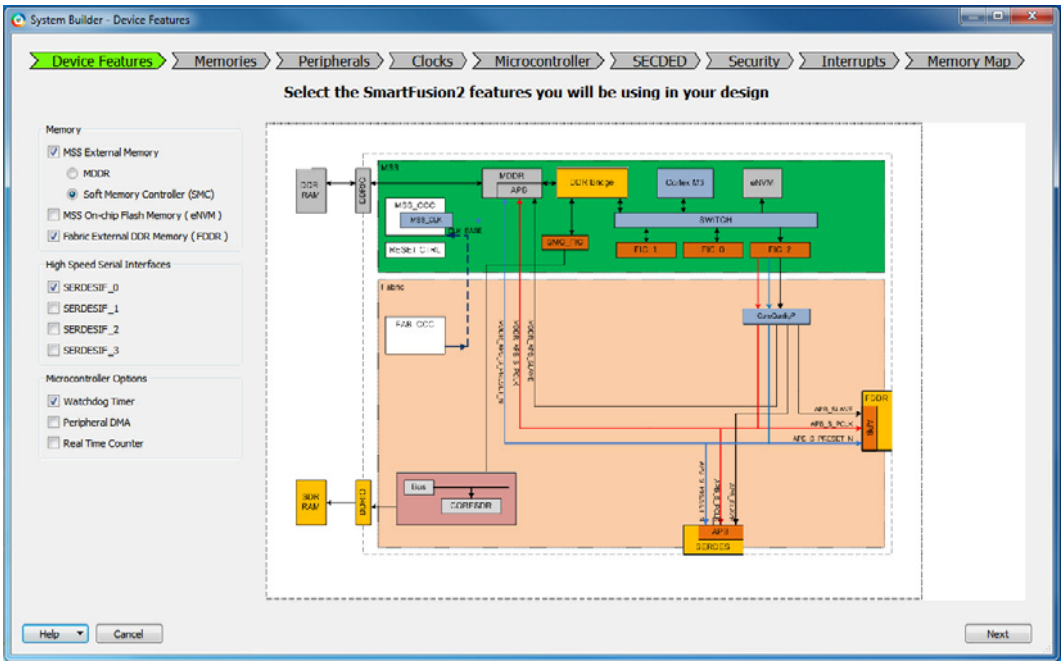

Figure 2-2 • **System Builder Device Features Page**

To enable access to an External Memory (e.g., LPDDR/DDR2/DDR3 RAM or SDRAM) from the MSS, select either MDDR or SMC:

- **MDDR** - This option allows access to an external DDR Memory via the MDDR (MSS DDR Controller). Select this option when you want to:
  - Access a DDR memory using the hardwired direct connection from the Cortex-M3 (or other MSS masters)

  AND/OR
  - Access a DDR memory from a master in the FPGA Fabric

  If you select this option, you need to specify your DDR Memory parameters in the "Memories Page".
- **Soft Memory Controller (SMC)** - This option allows the MSS to access a Soft Memory Controller in the Fabric. Select this option when you want to instantiate your own Soft Memory Controller in the FPGA Fabric, and access it from the Cortex-M3 or other MSS masters.

  A Soft Memory Controller (CORE_SDR_AXI) is automatically instantiated in the Fabric to control a Single Data Rate external Memory (SDR). You may want to select this option if you have a non-DDR based external memory chip that you want to interface to.

Note: You can select either MDDR or SMC but not both.

- **MSS On-Chip Flash Memory (eNVM)** - If you choose eNVM, you need to configure the ENVM in the Memories page. The eNVM is used for storing data clients or initialization clients (for programming).
- **Fabric External DDR Memory (FDDR)** - This option allows access to an external DDR Memory (LPDDR/DDR2/DDR3) via the FDDR (Fabric DDR Controller). Select this option when you want to access an external DDR memory from a master in the FPGA Fabric.

If you select this option, you need to configure the FDDR in the "Memories Page".

High Speed Serial Interfaces

Depending on the device you have selected for your Libero project, you may have zero to four SERDES blocks available for your system. Use the check box to select which SERDES blocks you will be using in your design. System Builder does not instantiate the SERDES blocks in the generated design but exposes the APB Master port for connection to each selected SERDES block. Use SmartDesign to instantiate the SERDES block at the top level of your design and connect to the appropriate ports in System Builder.

Refer to UG0447: SmartFusion2 and IGLOO2 High Speed Serial Interfaces User's Guide for a complete description of all the SERDES options and registers.

Microcontroller Options

You can select the following Microcontroller options if you want them enabled in your design. If unselected, they will be disabled.

- Watchdog Timer
- Peripheral DMA
- Real Time Counter

Memories Page

If your design uses external LPDDR/DDR2/DDR3 memories, you need an FDDR or an MDDR controller configured properly to work with your external DDR memory. If you want to use the on-chip eNVM to control a data storage client or a serialization client (for unique device information during programming), the eNVM needs to be configured as well.

The Memories page is where you configure the MDDR/FDDR/eNVM that you have previously specified as Device Features.

The MDDR/FDDR contains control registers, the value of which must be set correctly to match the parameters of the external DDR memories. When you configure the FDDR/MDDR in this page, System Builder takes your configuration settings and builds the correct FDDR/MDDR with the correct configuration register values set to work with your external DDR memory.

For eNVM configuration, you specify the client type, size and start and end address of your eNVM data storage client and/or Serialization client.

The Memories page has three tabs:

- FDDR
- MDDR
- ENVM

See Figure 2-3.

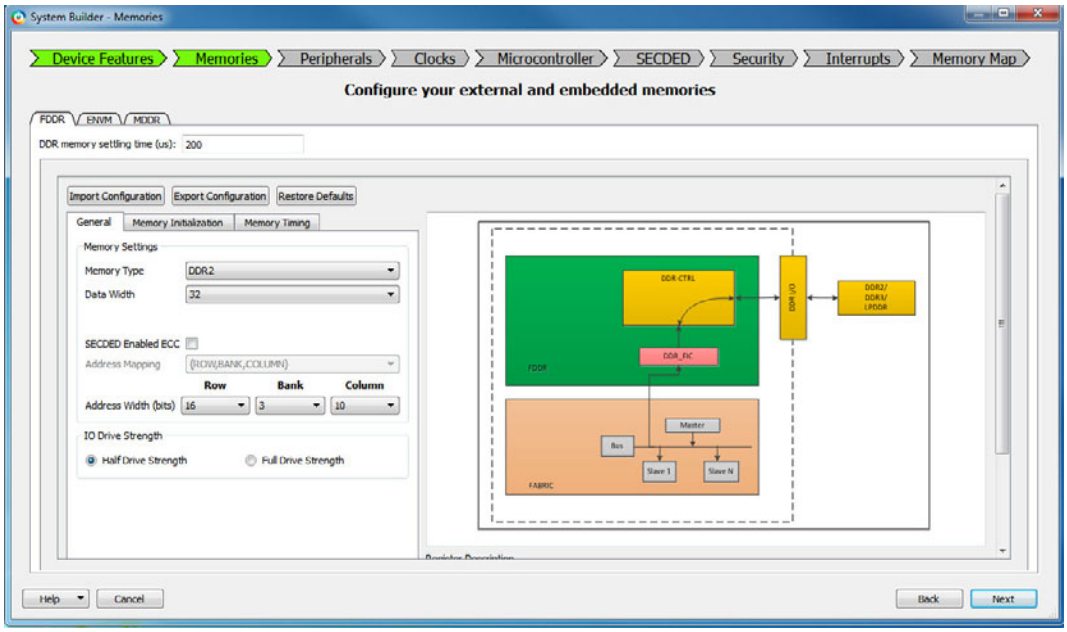

***Figure 2-3* • System Builder Memories Page**

MDDR/FDDR Configuration

If you have selected the MDDR or FDDR option in the Device Features page, you must configure the MDDR/FDDR controller to match the DDR memory parameters.

Refer to your DDR Memory Data Sheet when you configure the MDDR/FDDR Controllers.

For details, refer to the SmartFusion2 MSS DDR Controller Configuration User Guide and SmartFusion2 FPGA Fabric DDR Controller Configuration User Guide.

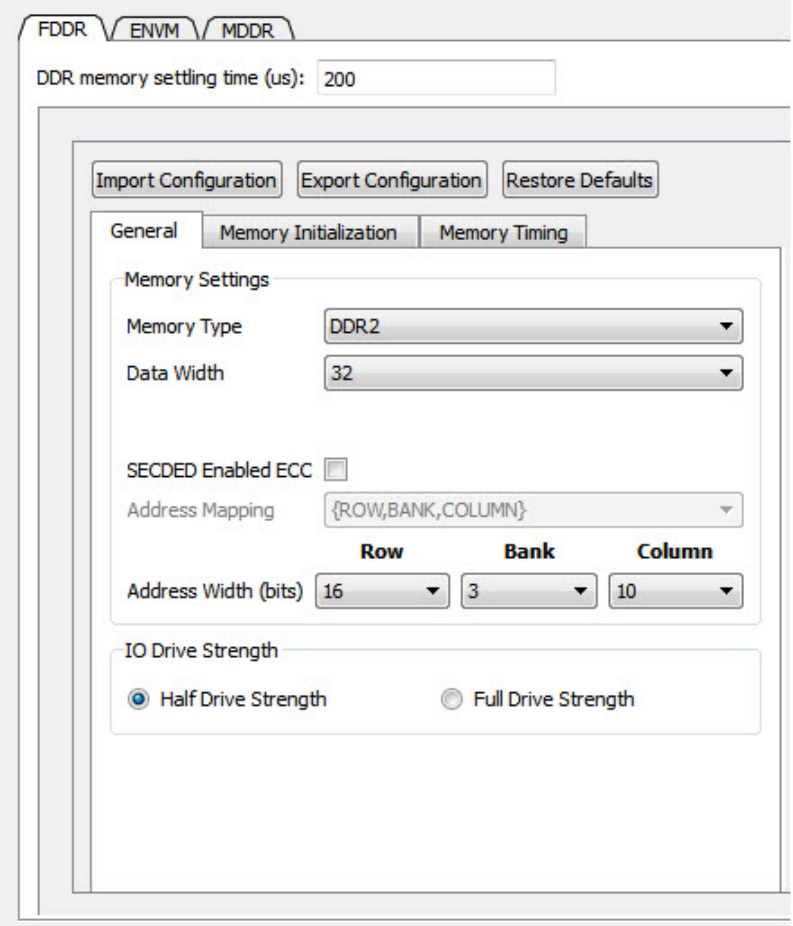

***Figure 2-4* • FDDR/MDDR Configuration**

MDDR/FDDR Memory Setting Time

This is the amount of time it takes for the DDR memory to be initialized. The default value is 200us. Refer to your DDR Memory Data Sheet for the correct value to use. An incorrect value may result in the DDR memory's failure to initialize.

Configurable DDR Memory Space (Row, Bank, Column)

The Memory space of the external DDR is configurable. Refer to the data sheet of your external DDR memory for the correct Row/Bank/Column settings. Note that the number for Row/Bank/Column refers to the number of Address Width (bits), not the absolute number of Rows/Banks/Columns. For example, if your external DDR memory has 4 banks, select 2 (2^2=4) for banks. If your external DDR memory has 8 banks, select 3 (2^3=8) for banks.

Import/Export Configuration File for MDDR/FDDR

The MDDR/FDDR configuration can be defined in a file and imported using the Import Configuration button. You can save your configuration and export it to a file using the Export Configuration button.

Refer to the SmartFusion2 High Speed DDR Interfaces User's Guide for a complete description of all the DDR configuration options and registers.

Embedded Flash (ENVM) Configuration

The ENVM tab lets you specify data storage and Serialization clients in the embedded Non-volatile Flash Memory (eNVM). Data storage clients are typically used to store your application firmware image or other types of data, such as DSP coefficients. For example. Serialization clients are used to add unique device information during programming.

From the Memories page, click the ENVM tab to access the eNVM Configurator. Select Data Storage or Serialization for the Client type and select Add to System (Figure 2-4). Click **Edit** to modify the configuration of your eNVM client.

Note: The number of available pages in the Flash Memory is device-dependent. Refer to your device data sheet for specific information on Flash Memory size.

Some eNVM pages are reserved to store the Certificate/Digest.

The number of Available Pages displayed in the eNVM Configurator is the total number of pages available to you after the Reserved Pages have been taken into account. For example, the M2S050 device data sheet shows a total of 2048 pages in the eNVM, but the eNVM Configurator (Figure 2-5) shows 2032 Available Pages, because 16 pages are reserved and are unavailable to the user.

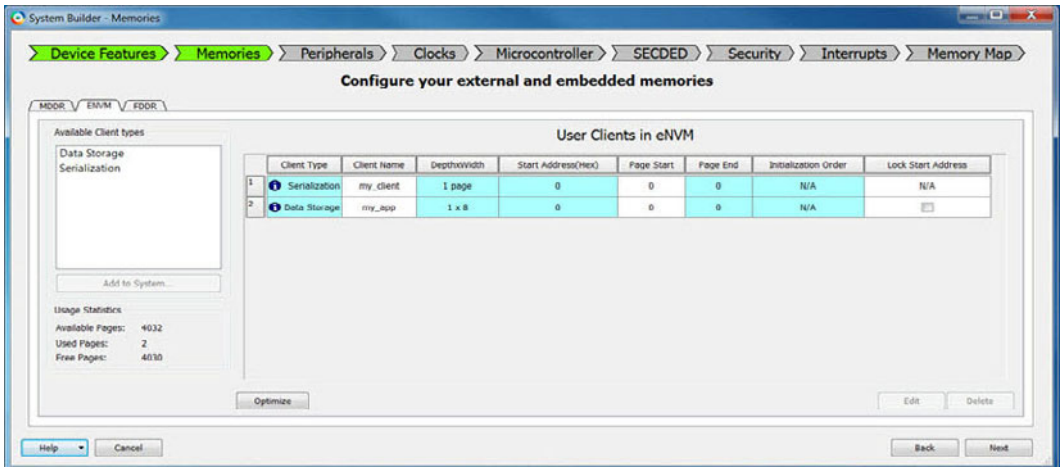

***Figure 2-5* • eNVM Configurator**

For details about the eNVM Configurator, refer to the SmartFusion2 MSS Embedded Nonvolatile Memory (eNVM) Configuration User Guide.

Peripherals Page

The Peripherals page allows you to build and configure various AMBA Master-Slave subsystems with simple drag-and-drop operations. A subsystem is defined as at least one master and at least one slave attached to an AMBA bus addressable from the MSS Microcontroller. The master can be a fabric master or the MSS Microcontroller. Just drag and drop the Slave and/or Master cores into the predefined subsystems you want to build.

For example, to build a fabric master-slave subsystem with the FIC_0 as the interface to the MSS, drag and drop an AMBA master and an AMBA slave into the MSS_FIC_0 Fabric Master Subsystem. You can configure these cores by clicking on the Configure icon after a core has been added to a subsystem.

Design rules checking (DRC) are enforced and tooltip messages appear to alert you when you make a wrong move. You need to correct the violations before you can continue. This ensures a correct-by-design system when you use System Builder.

In addition, this page allows you to enable/disable the Microcontroller (MSS) features such as UART, GPIO, Ethernet, MAC, CAN, USB, SPI and I2C

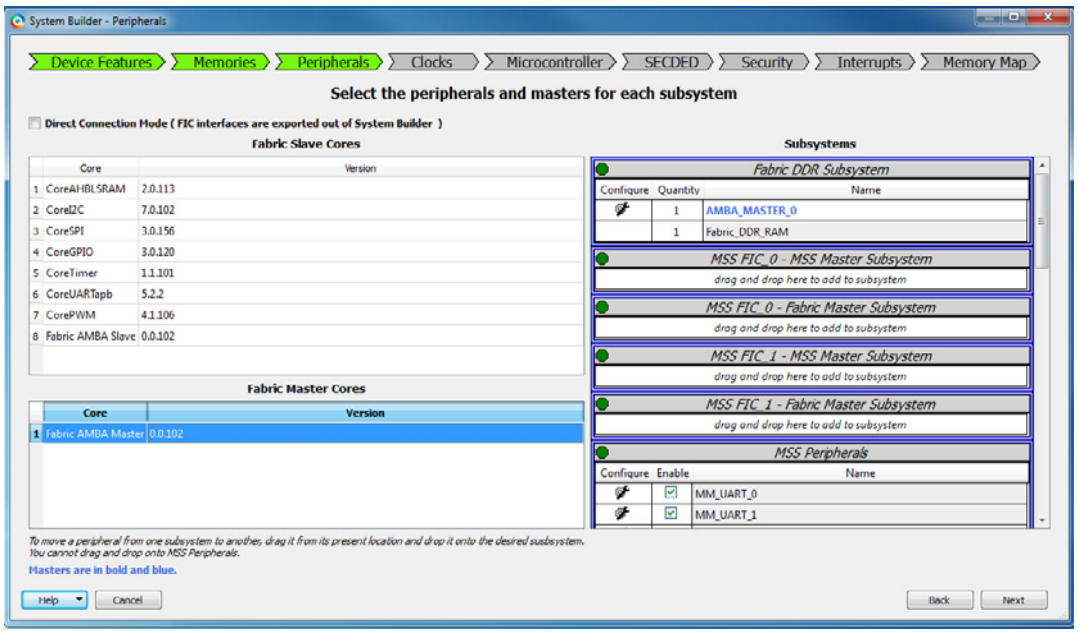

Figure 2-6 • **System Builder Peripherals Page**

The Peripherals page has four panels:

- Fabric Slave Cores Panel
- Fabric Master Cores Panel
- Subsystems Panel
- MSS Peripherals Panel

.

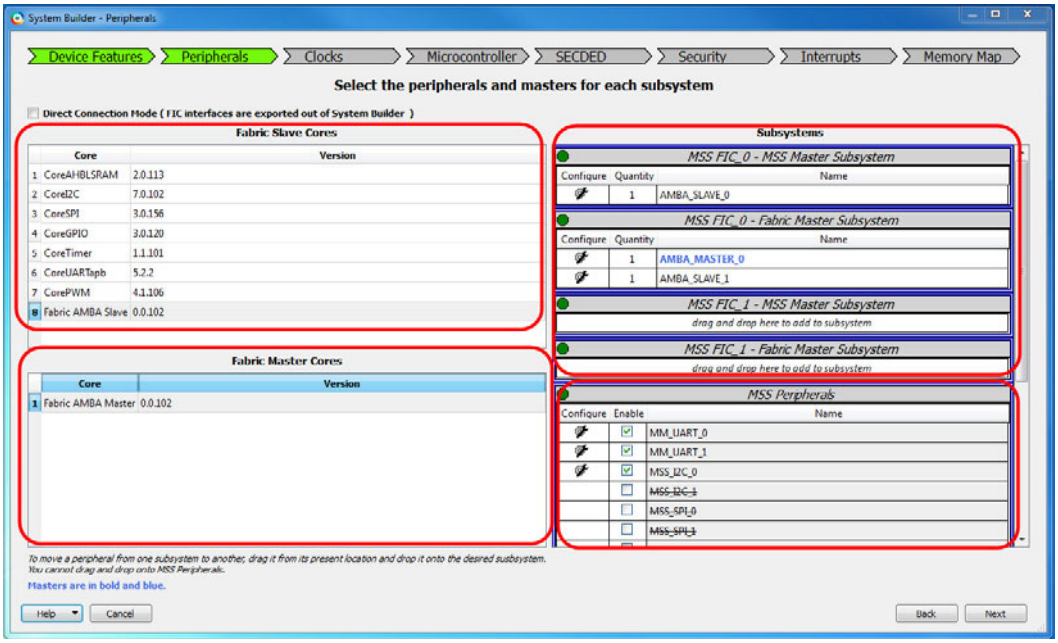

***Figure 2-7* • Select Peripherals - Fabric Slave, Fabric Master and Subsystems**

Fabric Slave Cores Panel

The Fabric Slave Cores panel contains a list of Fabric slave cores. Two types of slave cores are available:

- Special-purpose custom AMBA slaves such as CoreSPI, CorePWM, CoreUARTapb, for example. When added to a subsystem, System Builder automatically instantiates these cores into the generated netlist.
- Generic fabric AMBA slaves. You can configure the interface type - AXI, AHBLite, APB - for these generic cores as well as the number of interrupts (up to 8) that these cores generate. System Builder automatically create top level ports (AMBA interface, clocks, resets and interrupts) for these generic AMBA cores. You must instantiate and connect the actual slave to the System Builder component ports at the next level of hierarchy using SmartDesign.

Note: The Fabric Slave Cores panel is disabled in Direct Connection Mode. See "Default Mode and Direct Connection Mode" on page 15.

Fabric Master Cores Panel

The Fabric Master Cores panel contains a generic Fabric AMBA master core. You can configure the interface type—AXI, AHBLite, APB—for this generic master core. System Builder automatically creates top level ports (AMBA interface, clocks, resets) for this generic AMBA core. You must connect the actual master to the System Builder component ports at the next level of hierarchy using SmartDesign.

Note: The Fabric Master Cores panel is disabled in Direct Connection Mode. See "Default Mode and Direct Connection Mode" on page 15.

Subsystems Panel

The SmartFusion2 Microcontroller Subsystem (MSS) offers up to five different Fabric Interface Controllers (FIC) depending on the selected device. These interface blocks enable the MSS to interface with logic implemented in the FPGA fabric and vice versa. The System Builder allows you to create a subsystem for each of those interfaces:

- **DDR_FIC**. The DDR_FIC is available when you configure the MSS DDR block (MDDR) such that the external DDR memory can be accessed from an FPGA fabric master via an AXI or AHBLite interface.
- **SMC_FIC**. The SMC_FIC is used when you configure the MSS DDR Block in the Single Data Rate (SDR) mode. In this configuration, the MSS accesses external Single Data Rate DRAM or Asynchronous memories via a soft memory controller instantiated in the FPGA fabric such as CoreSDR_AXI and CoreSDR_AHB. The

SMC_FIC is an AXI or AHBLite slave AMBA interface. The DDR_FIC and SMC_FIC interfaces are mutually exclusive; only one is active at a time, depending on your selection in the Device Features page.

- **FIC_0 Fabric Master.** The FIC_0 master interface enables you to naturally extend the MSS AMBA bus into the FPGA fabric.
- **FIC_0 MSS Master**. Using this interface you can have a master in the fabric accessing the MSS internal peripherals.
- **FIC_1 Fabric Master.** The FIC_1 master interface enables you to naturally extend the MSS AMBA bus into the FPGA fabric. This interface is not present on all devices.
- **FIC_1 MSS Master**. Using this interface you can have a master in the fabric accessing the MSS internal peripherals. This interface is not present on all devices.

Note: The MSS FIC_2 interface is not available to you (as user), as it is reserved by System Builder to create the peripheral initialization configuration bus for the SERDES Block, the MDDR and FDDR blocks.

Figure 2-8 shows the MSS and Fabric Interface Controllers (FICs).

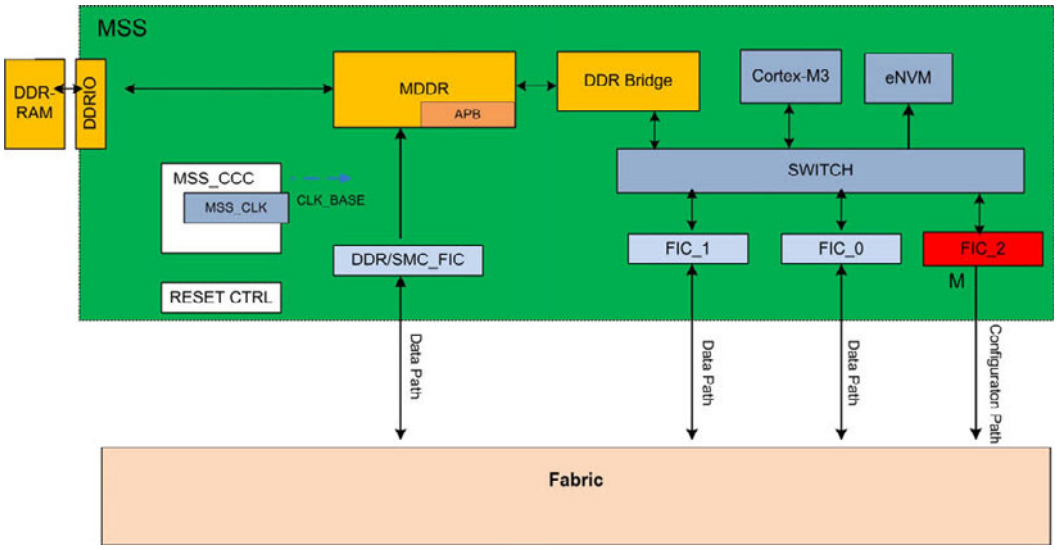

Figure 2-8 • **MSS and Fabric Interface Controllers**

Each Fabric Interface Controller and the associated subsystem can operate on a different clock frequency, defined as a ratio of the MSS main clock M3_CLK. The SmartFusion2 architecture imposes a certain number of rules related to clocking domains between the Fabric Interfaces and the FPGA Fabric. System Builder accounts for these rules and generates a correct-by-construction design based on your inputs.

When you drag the available cores into a subsystem (in the Subsystems panel) and configure these cores into a subsystem, the correct bus interface pins, clocks, and resets are exposed at the top level System Builder generated design. At the top level SmartDesign, you can then connect these ports to your custom peripheral(s).

For generic Slave Cores, you can click the wrench icon to configure the AMBA interface type (one of AXI, AHBLite or APB) and the number of Interrupts (up to 8) per slave to the MSS.

.

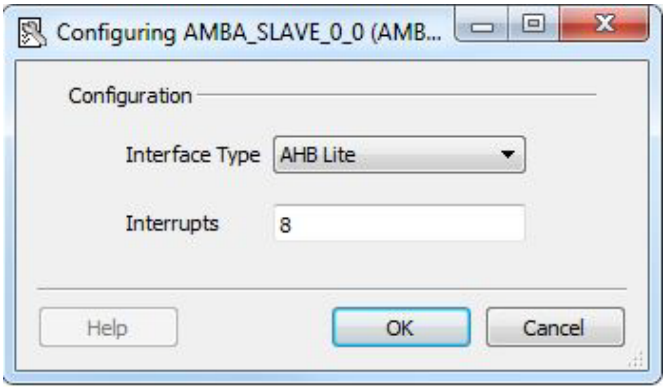

***Figure 2-9* • Generic AMBA Slave Configuration - Interface Type and Interrupts**

You can add/delete/move masters and peripherals around the subsystems to define how you want the various masters and peripherals to communicate.

To delete a peripheral from a subsystem; right-click and choose **Delete**.

MSS Peripherals Panel

This panel allows you to enable/disable the MSS Microcontroller silicon features such as UART, I2C, SPI, GPIO, USB, MAC and CAN. Check the Enable check box to enable the Peripheral and then double-click the wrench icon to configure the options.

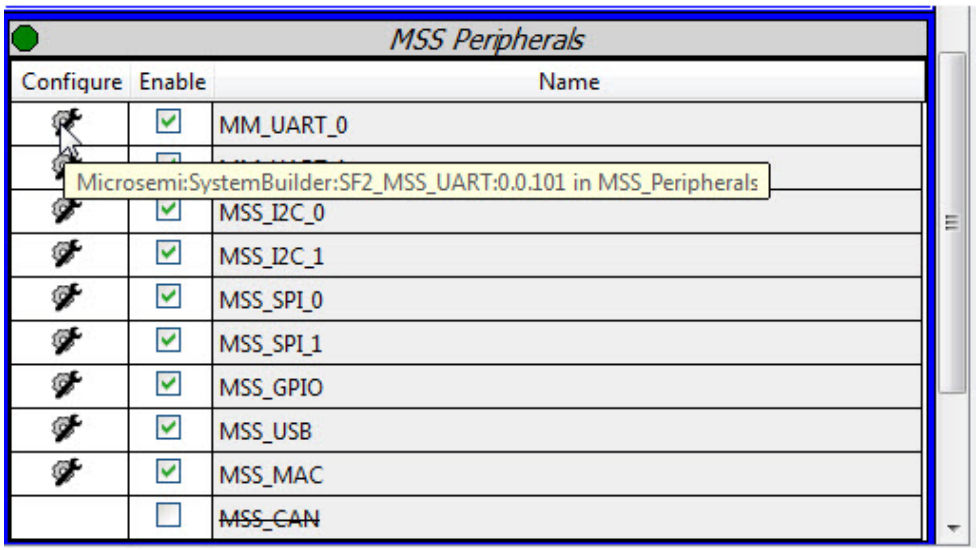

***Figure 2-10* • Generic AMBA Slave Configuration - Interface Type and Interrupts**

Default Mode and Direct Connection Mode

System Builder operates in two modes:

- **Normal (Default) Mode** - In this mode, System Builder builds the AMBA bus structure (buses and bridges) of the correct AMBA type and attaches the Masters and Slaves to the bus cores based on your configuration and specifications. In addition, System builder also builds the Clock network, the Reset circuitry and the Interrupt circuitry for every subsystem. This is the default mode.

- **Direct Connection Mode** - In this mode, System Builder does not build the bus structure for each subsystem. It merely exposes the Master/Slave BIF (Bus Interface) ports to the top level. You need to construct your own bus structure outside of System Builder and connect the Master/Slave slots of the Bus you have built to the exposed BIF ports of the System Builder block. In this mode, System Builder builds only the Clock network and the Reset Circuitry but not the bus structure. This is the non-default mode. To enable this mode, check the Direct Connection Mode check box in the Peripherals page.

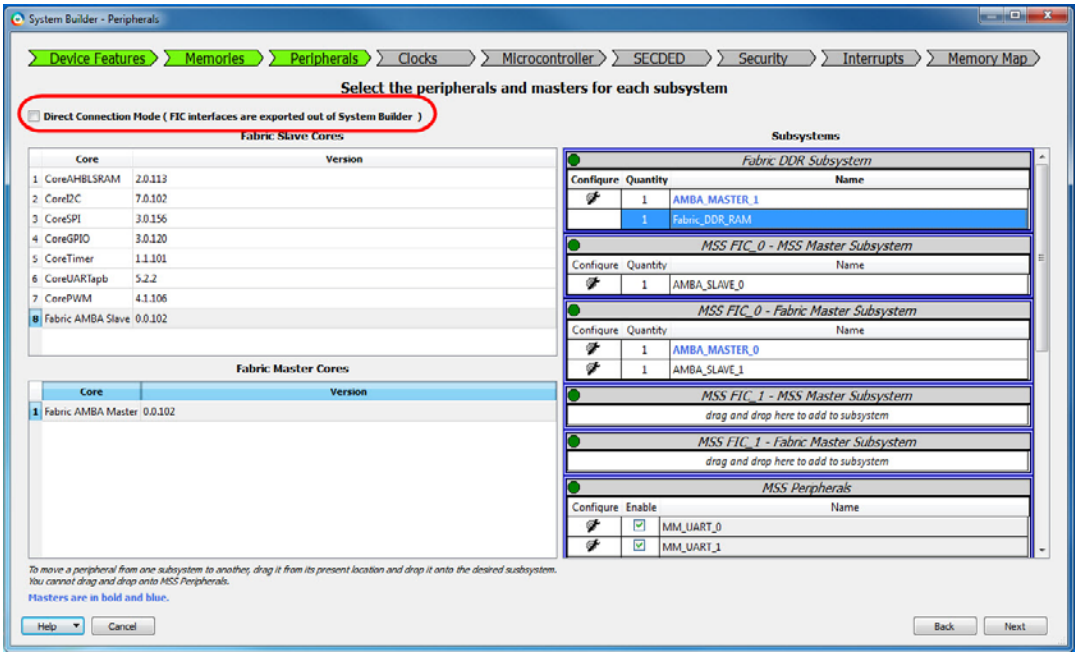

***Figure 2-11* • Direct Connect Mode Option**

A message box appears to alert you how System Builder will operate in the Direct Connection Mode. Click **Yes** to continue.

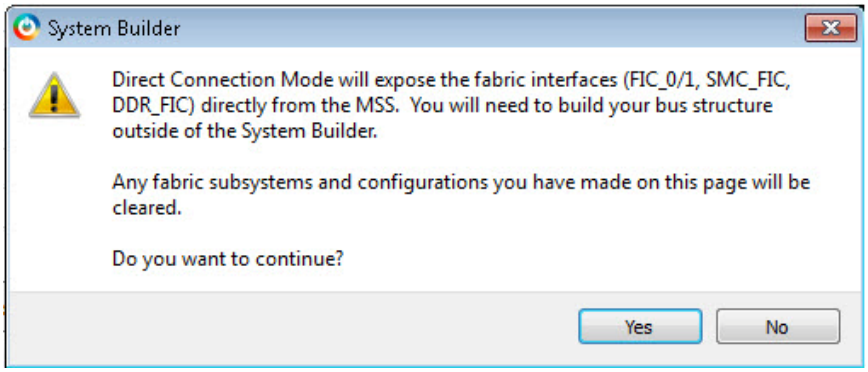

***Figure 2-12* • System Builder Direct Connection Mode Alert Message**

Why and When to use Direct Connection Mode

Use the Direct Connection Mode if and when any one of the following applies:

- You want System Builder to construct the Clock Tree, Reset and Interrupt Circuitry, but not the bus structure, of your system block and then pass this system block as a golden blackbox to another design team for the construction of the slave logic and the final hook-up of the slave logic to the System Builder block. Direct

Connection Mode is the only solution because it does not require you to specify up front the number and type of slaves for your system.

- You want to directly connect (wire connection) the master/slave BIF coming from the MSS or FDDR to your master/slave BIF, without involving a bus structure in between.
- You want to expose an AXI FIC from the MSS (MSS_SMC_AMBA_MASTER configured as AXI) and use this AXI Master BIF to achieve higher throughput transactions from the MSS to the fabric.
- You want the flexibility to choose the slot size (address width) and the slave slot numbers of the bus cores.
- You want the flexibility to choose the bus/bridge core version.

Example - FIC_0_MSS_MASTER Subsystem in Normal vs Direct Connection Mode

This example creates a FIC_0 MSS Master subsystem. In the normal (default) mode, System Builder builds the bus core (CoreAHBLite/CoreAXI bus core) and connects the FIC_0_AHB_MASTER Bus Interface (BIF) to the Master Port (M0) of CoreAHBLite bus (Figure 2-13 below).

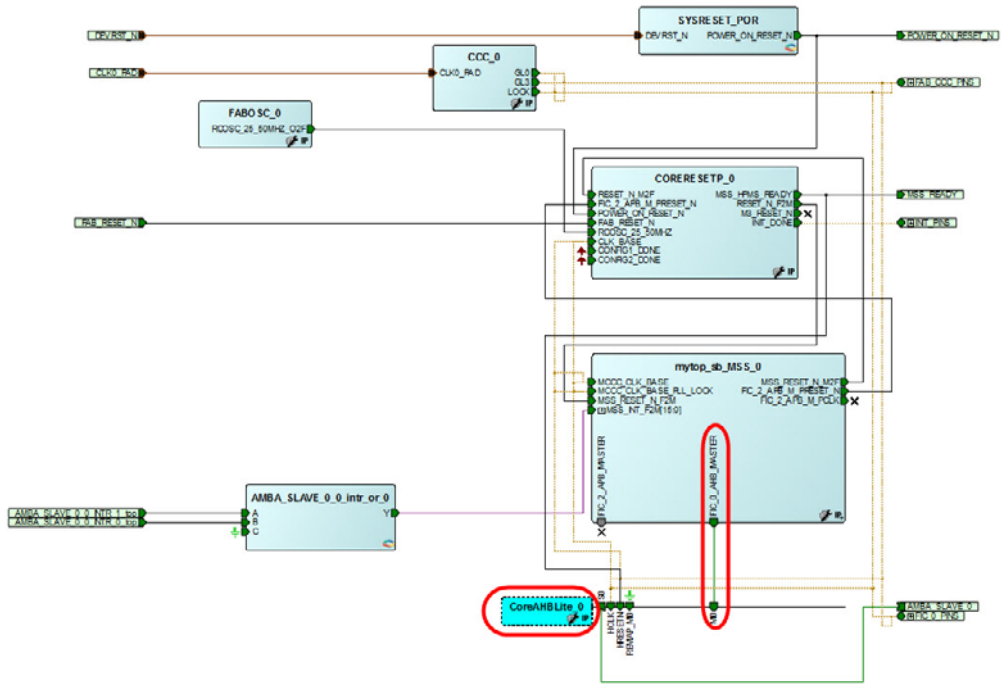

***Figure 2-13* • System Builder Default Mode (CoreAHBLite/Core AXI Core created)**

In the Direct Connection (non-default) Mode, however, System Builder does not build the CoreAHBLite/CoreAXI Bus core. It merely exposes the FIC_0_AHB_MASTER Bus Interface port to the top level (Figure 2-14 below). You need to build your own bus core of the correct AMBA type (AHBLite) outside System Builder and connect the FIC_0_AHB_MASTER Bus Interface (BIF) port to the Master slot of the bus core (CoreAHBLite) you have created outside of System Builder.

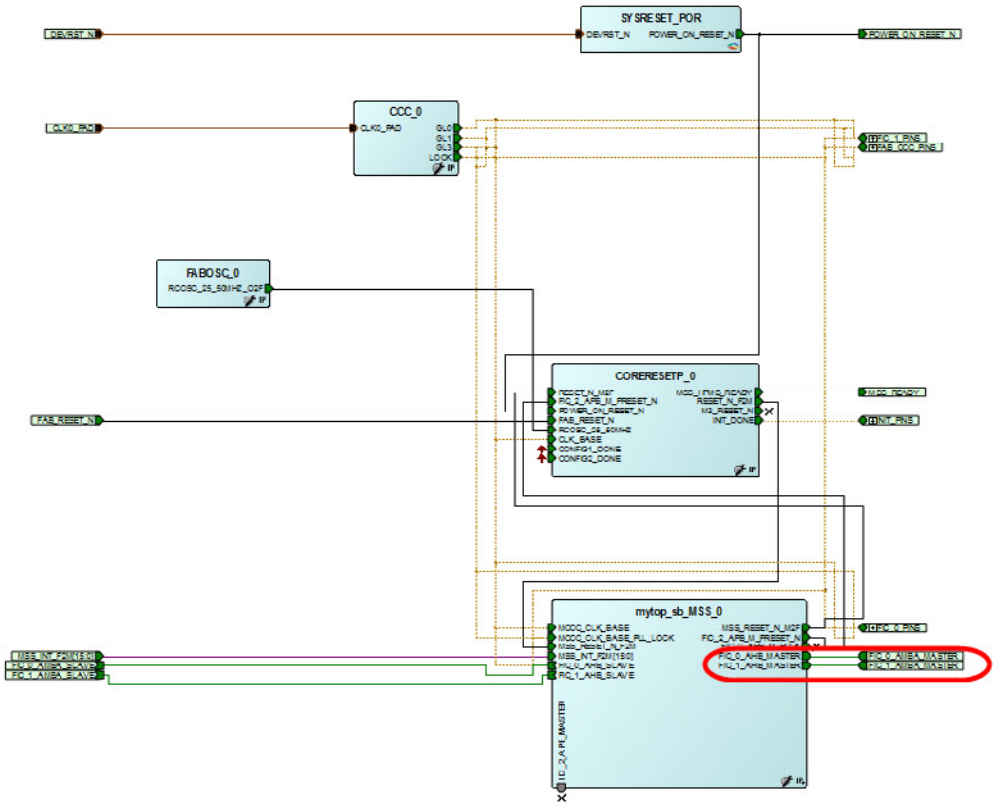

Figure 2-14 • **System Builder Direct Connection Mode (No Bus Core created and) AMBA Master BIF Promoted to Top Level)**

Configuring a Subsystem in Direct Connection Mode

The Direct Connection Mode is a global option. This option cannot be applied on a per-subsystem-basis. It applies to *all* the subsystems that are enabled as per your specifications and configuration in the Device Features page:

- FIC_0/1 - MSS Master Subsystems
- FIC_0/1 - Fabric Master Subsystems
- MSS DDR FIC Subsystem
- MSS SMC FIC Subsystem
- Fabric DDR Subsystem

When Direct Connection Mode is enabled, you can check/uncheck the check boxes next to the master/slave cores in each subsystem to choose which master and slave bus interfaces from the MSS and/or FDDR blocks you would want to be exposed on the top level System Builder design. You can also click on the wrench icon next to each of the cores and configure the AMBA type of each of the master and slave interfaces that will be exposed on the top level System Builder design. Proper Design Rule Checking (DRC) has been implemented in each of the subsystems ensuring you to build correct-by-construction designs in System Builder.

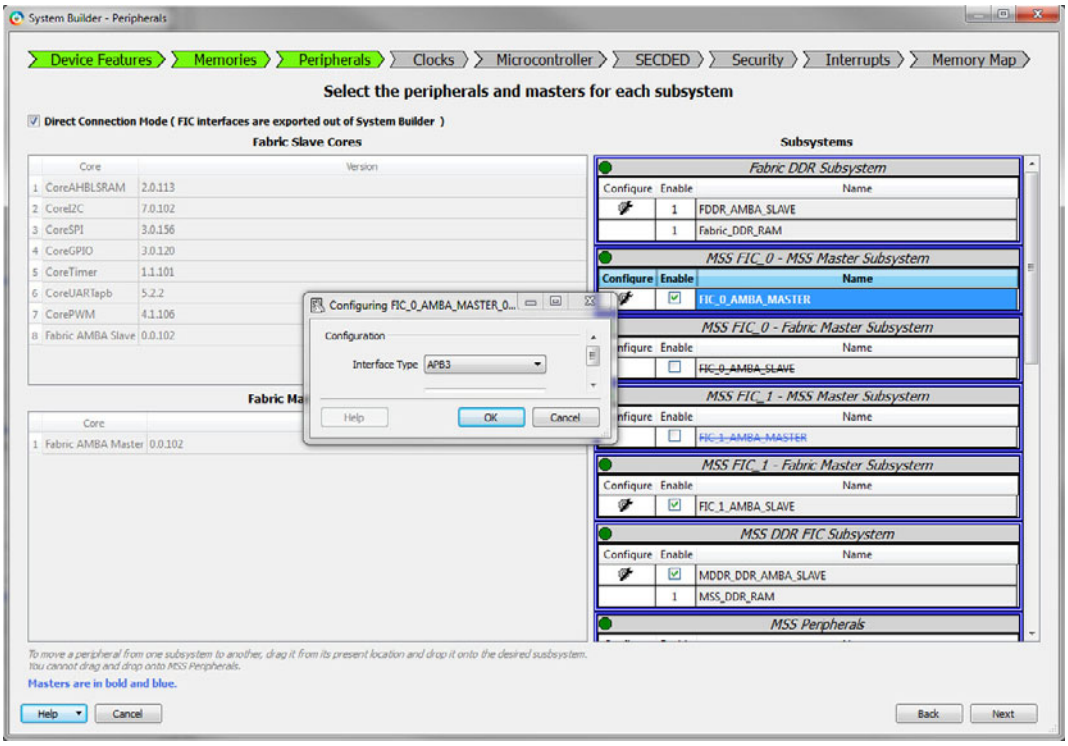

Figure 2-15 • **Configuring AMBA Type**

Note: When Direct Connection Mode is enabled, the "Fabric Slave cores" panel and the "Fabric Master cores" panel are disabled/grayed out. You cannot drag and drop any master/slave cores from these panels into any subsystem. You need to build (instantiate and hook-up) the master and slave cores outside the System Builder block when operating in the Direct Connection Mode.

Follow-up Actions after Generating a Design in Direct Connection Mode

In Direct Connection Mode, the System Builder block exposes to the top level all Master/Slave Bus Interfaces (BIF) corresponding to the subsystems that are enabled as per your specifications and configuration in the Device Features page and the Peripherals page. After System Builder generation completes, you need to:

1. Manually build the correct AMBA-compliant bus-bridge core for every subsystem
2. Connect the master/slave BIF ports exposed on the System Builder block to the appropriate BIF ports of the bus cores.
3. Connect the clock and reset pins of the slave/master and bus cores to the appropriate clock and reset pins of the corresponding subsystem pin groups.

4. Instantiate your actual master and slave cores in the design and connect all their clock, reset, interrupt and bus interface ports appropriately.

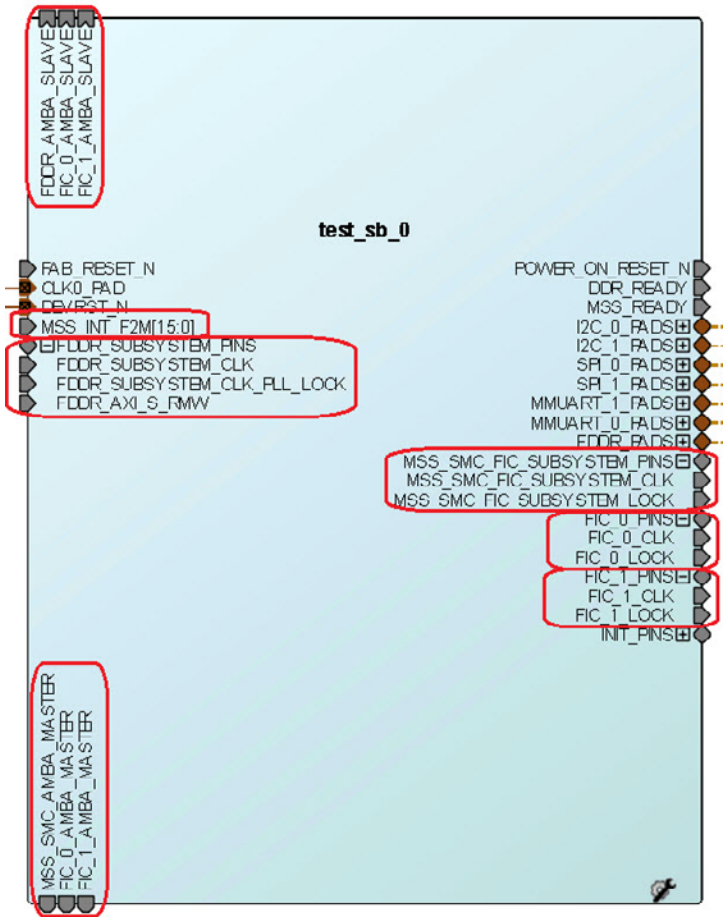

***Figure 2-16* • System Builder Block and Top-Level Ports (Direct Connection Mode)**

Interrupt Management for Direct Connection Mode

When operating in default mode, if fabric slave cores with interrupt pins are added to the MSS FIC_0/1 - MSS Master Subsystems, System Builder manages the interrupt pins of the slave cores by configuring the MSS Interrupt Management configurator and connecting the interrupt pins of the slave cores to the MSS interrupt bus - MSS_INT_F2M[15:0]. You can further view and modify the interrupt pin assignments in the System Builder Interrupts page.

In Direct Connection Mode, System Builder will not manage the interrupts from the fabric slave cores to the MSS because the number and type of slaves and their interrupts are not specified while the design is being constructed in System Builder. However, in this case, System Builder will expose the MSS interrupt bus - MSS_INT_F2M[15:0] on the top level System Builder design. You will have to connect the interrupt pins of the fabric slave cores (that are hooked up to the MSS FIC_0/1 - MSS Master Subsystems) to the sliced bits of MSS interrupt bus - MSS_INT_F2M[15:0].

Clocks Page

The System Builder Clocks page enables you to enter and configure clock parameters for the clocks that drive various subsystems. Use this page to configure how fast you want your subsystem clocks to run. It allows you to configure the Clock source, the clock frequency, the Clock Conditioning Circuit (CCC) and the Oscillators for your subsystems.

Based on the configuration settings you make on this page, System Builder constructs the clock tree for your top level System Level Block. Three tabs are available in the Clocks page:

- Clock
- Fabric CCC
- Chip Oscillators

Clock Tab

The Clock tab (Figure 2-6) enables you to select the system level clock source, the Cortex M3 clock frequency as well as the clock frequencies of your subsystems. System Builder automatically instantiates and configures the required CCC to generate the subsystem clocks.

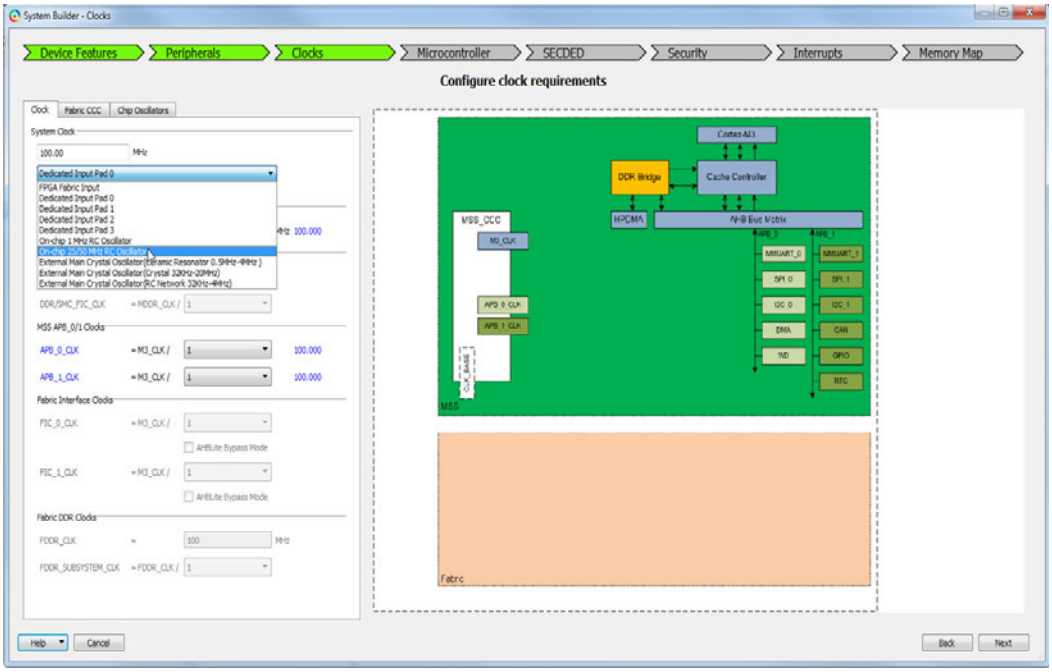

Figure 2-17 • **System Builder Clocks Page - Clock Tab**

Your system clock source options include:

- FPGA Fabric Input
- Dedicated Input Pad 0/1/2/3

  These are dedicated pads for clock source routed on a clock global network. If your design has many globals or you have hardware routed on the board to connect to CLK<1/2/3>_PAD, you have the flexibility to pick which CLK_PAD you want to use for your system clock.
- On-chip 1 MHz RC Oscillator
- On-chip 25/50 MHz RC Oscillator (50MHz for 1.2V devices and 25MHz for 1.0V devices)
- External Main Crystal Oscillator (Ceramic Resonator 0.5MHz to 4MHz)
- External Main Crystal Oscillator (Crystal 32KHz - 20MHz)
- External Main Crystal Oscillator (RC Network 32KHz - 4MHz)

You must also specify your clock source frequencies if it is not predefined by the clock source option. All other clocks in the system are derived from the System Clock settings. Choose a multiplier or divisor value from the pull-down list to configure the clock frequency for your subsystems. The tool enforces clock frequency dependency rules. Invalid frequencies will be flagged as errors.

Cortex-M3 and MSS Main Clock - Specifies your Cortex-M3 frequency.

MDDR Clocks - Consists of two clock domains: MDDR_CLK and DDR/SMC_FIC_CLK. The MDDR_CLK specifies your external DDR memory frequency. The DDR/SMC_FIC_CLK specifies the frequency at which your fabric logic interfaces with the MDDR controller.

MSS APB_0/1 Clocks - The APB_0_CLK and APB_1_CLK is the frequency of the MSS Peripherals on the APB_0 and APB_1 buses.

Fabric Interface Clocks - The FIC_0_CLK is the frequency at which the MSS FIC_0 subsystems will operate. The FIC_1_CLK is the frequency at which the MSS FIC_1 subsystems will operate.

AHBLite Bypass Mode - Use this option to enable the FIC Bypass Mode. In this mode, signals to and from the fabric are not registered, thus fewer clock cycles are required to complete each transaction but the overall system frequency may be lower than what could be achieved in pipelined mode (non-bypass mode).

This option is for the AHBLite interface type only. The clock ratio between M3_CLK and FIC_0/1_CLK must be set to 1:1 when the AHBLite Bypass Mode is selected. This clock ratio requirement is enforced in the MSS CCC Configurator when this mode is selected.

Fabric DDR Clocks - The FDDR_CLK specifies your external fabric DDR memory frequency. The FDDR_SUBSYSTEM_CLK is the frequency at which your fabric logic interfaces with the Fabric DDR controller.

Fabric CCC Tab

The Fabric CCC tab (Figure 2-18) enables you to select additional fabric CCC resources not used by any of your configured subsystems. These Fabric CCC clocks are grouped under FAB_CCC_PINS group and promoted to the top level of the System Builder block. Configure these clocks and use them to drive your own fabric logic. Depending on what subsystems you have configured, there may be up to four additional FAB_CCC clocks:

- FAB_CCC_GL0
- FAB_CCC_GL1
- FAB_CCC_GL2
- FAB_CCC_GL3

If one or more of the above pins are promoted and exposed under the FAB_CCC_PINS group, then another output pin FAB_CCC_LOCK is also promoted and exposed under the same pins group. This is the LOCK signal of the fabric CCC.

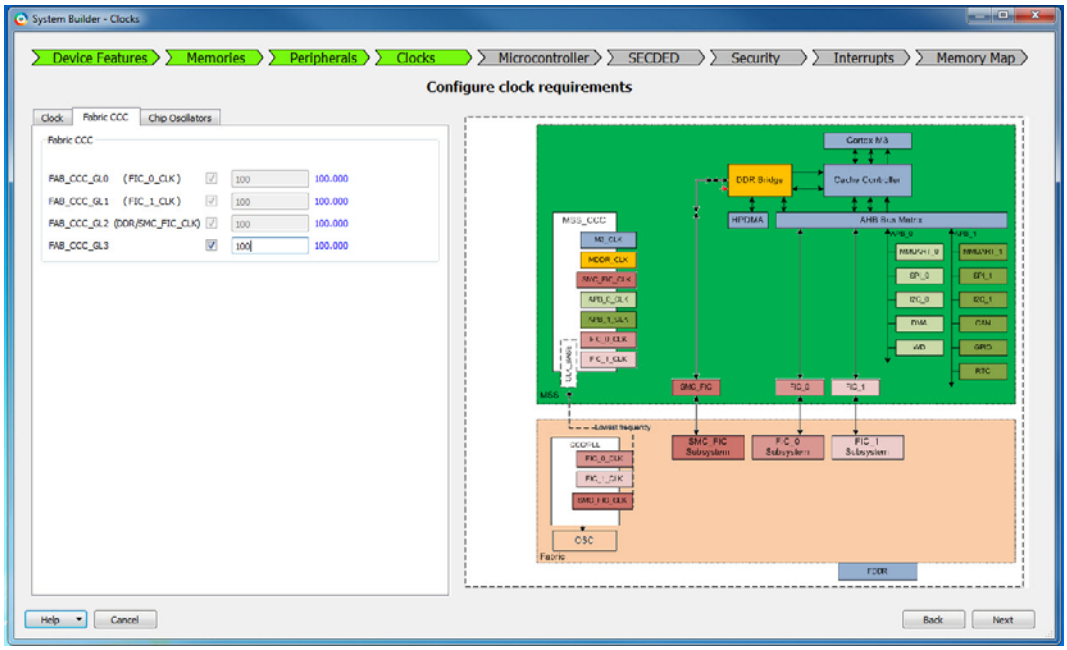

***Figure 2-18* • System Builder Clocks Page - Fabric CCC Tab**

Chip Oscillators Tab

The Chip Oscillators tab shows all the physical clock oscillator resources (Figure 2-19). If you want to use a Fabric Oscillator to drive the Fabric CCC or the Fabric Logic, check the appropriate checkbox. System Builder exposes/promotes the appropriate pins to the top level under the CHIP_OSC_PINS group.

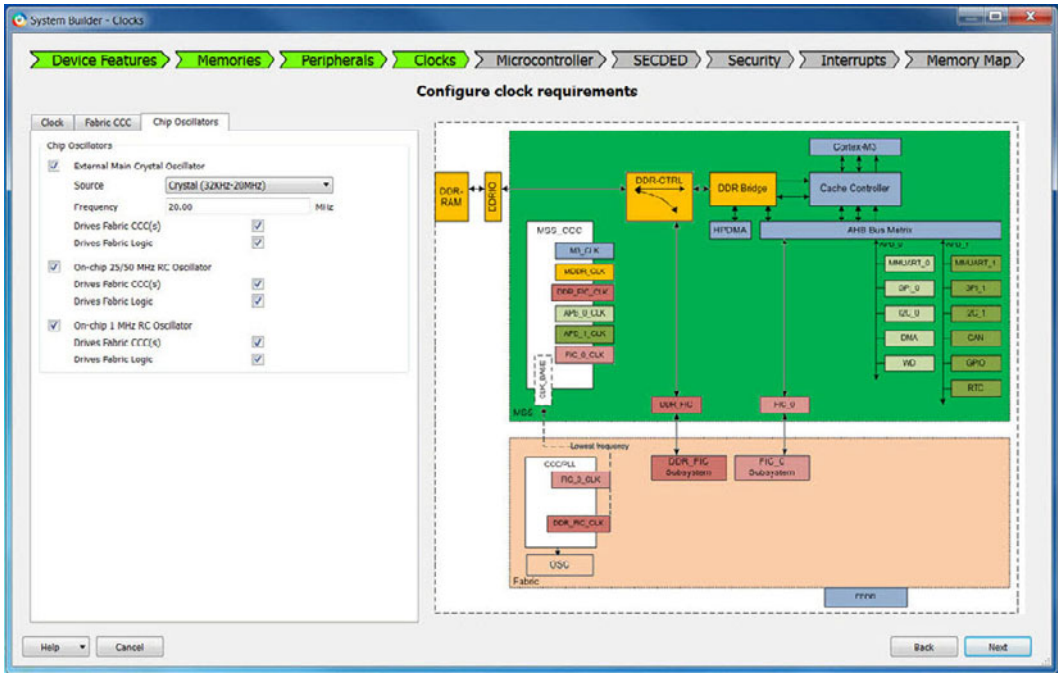

***Figure 2-19* • System Builder Clocks Page - Chip Oscillators Tab**

Note: In the Clock tab of the Clocks page, if the System Clock's source is configured to be any of the External Main Crystal Oscillator options (Ceramic/Crystal/RC Network), the Source and Frequency fields under the External Main Crystal Oscillator section in the Chip Oscillators tab are automatically configured and are grayed out. You can check the 'Drives Fabric CCC(s)' and 'Drive Fabric Logic' checkboxes to expose the corresponding pins to the top level.

Microcontroller Page

This page allows you to configure the Cortex-M3 Microprocessor and its Cache Controller. The MSS AHB Bus matrix, Watchdog Timer, Peripheral DMA, Real Time Counter inside the MSS are also configured on this page.

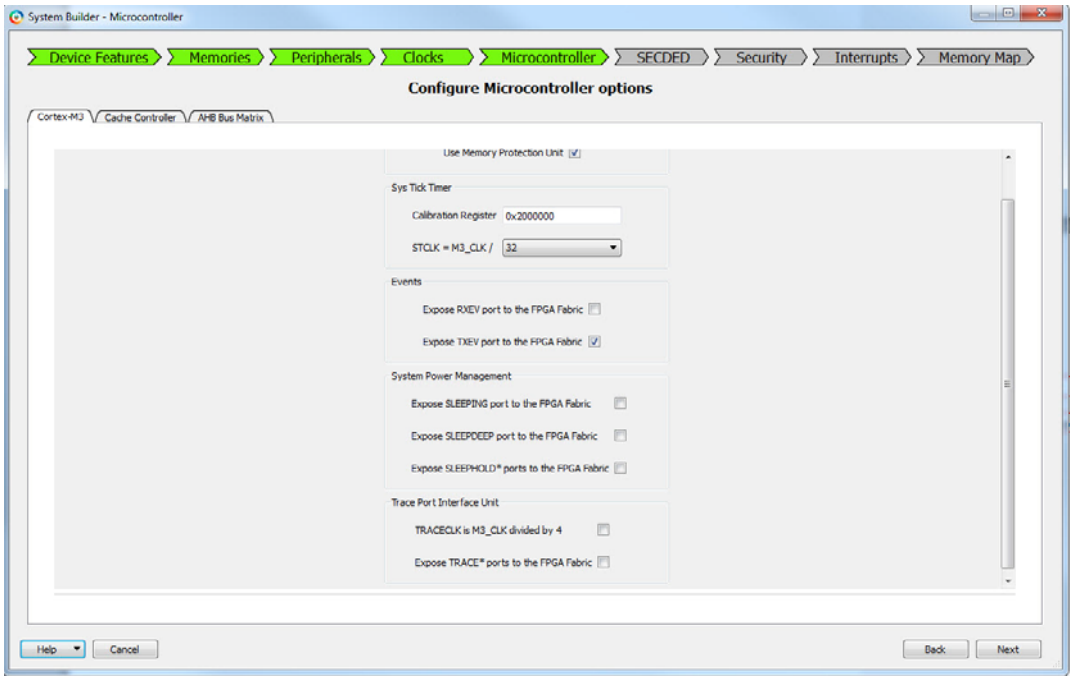

Figure 2-20 • **System Builder Microcontroller Page**

For details on how to configure the Microcontroller, refer to the following:

- SmartFusion2 MSS ARM Cortex-M3 Configuration
- SmartFusion2 MSS Cache Controller Configuration
- SmartFusion2 MSS AHB Bus Matrix Configuration

SECDED Page

The SECDED (Single Error Correct Double Error Detect) Configuration page allows you to enable/disable EDAC (Error Detection and Correction) on certain MSS peripherals such as CAN, Ethernet, USB, eSRAMs and specify the conditions under which an EDAC interrupt is generated (1-bit error, 2-bit error, both or none).

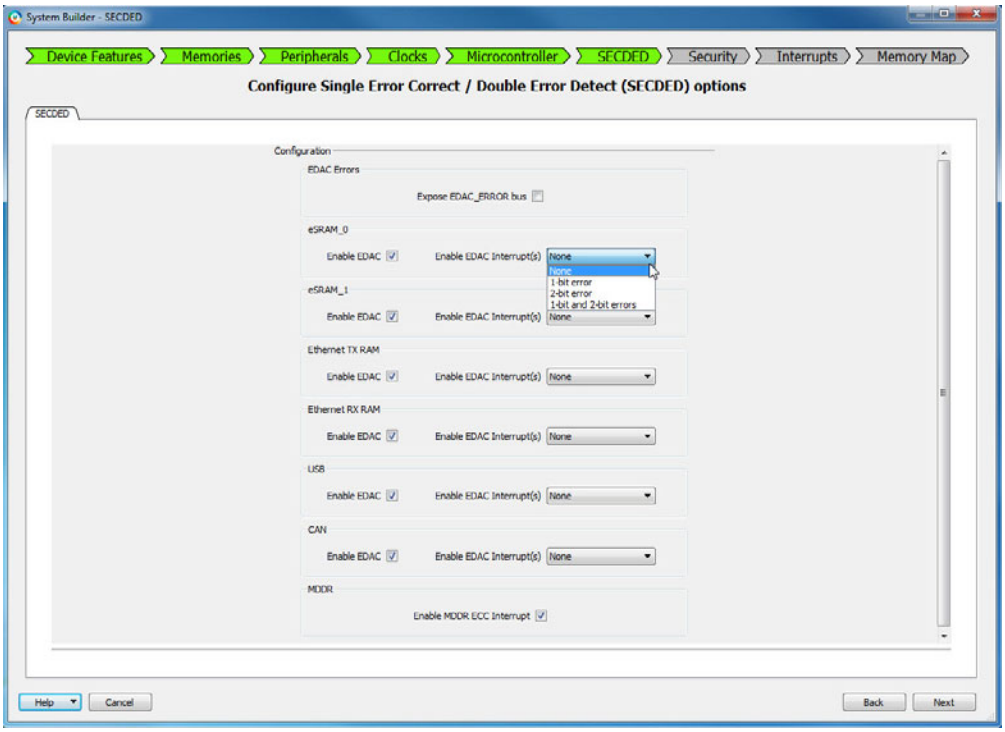

Figure 2-21 • **System Builder SECDED Page**

For details, refer to the SmartFusion2 MSS Single Error Correct/Double Error Detect Configuration User Guide.

Security Page

The System Builder Security page allows you to configure the read and/or write access restrictions to the MSS peripherals such as eSRAMs, eNVMs, FIC_0/1, DDR Bridge.

The Security page also allows you to control access to the MSS memory map from fabric masters. The Security page is configurable only for devices that support advanced security denoted by the suffix "S" in the die name.

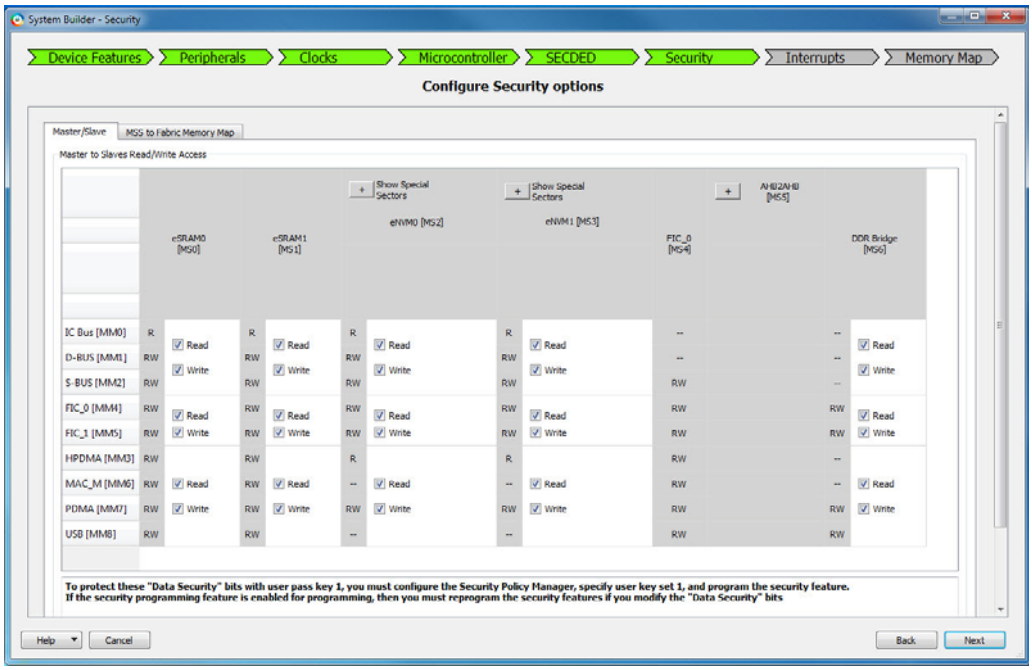

***Figure 2-22* • System Builder Security Page**

For details about how to configure the Security, refer to the SmartFusion2 MSS Security Configuration User Guide

Interrupts Page

When you step through this page, System Builder generates the interrupt circuitry based on the interrupt options you have made on the Peripherals page when you configure the AMBA slaves in your subsystems. You may use this page to change or lock the pin assignment of Interrupt signals.

MSS Interrupts

The MSS has 16 dedicated Interrupt lines from the Fabric. When there are 16 or fewer slaves with interrupts, the Interrupt pins of the fabric slaves 0 through 15 are connected to the MSS_INT_F2M[15:0] pins via cascaded OR-gates. In this case, each slave has a dedicated Interrupt access to the MSS. (Figure 2-23).

Figure 2-23 shows 16 slaves with dedicated interrupt access to MSS.

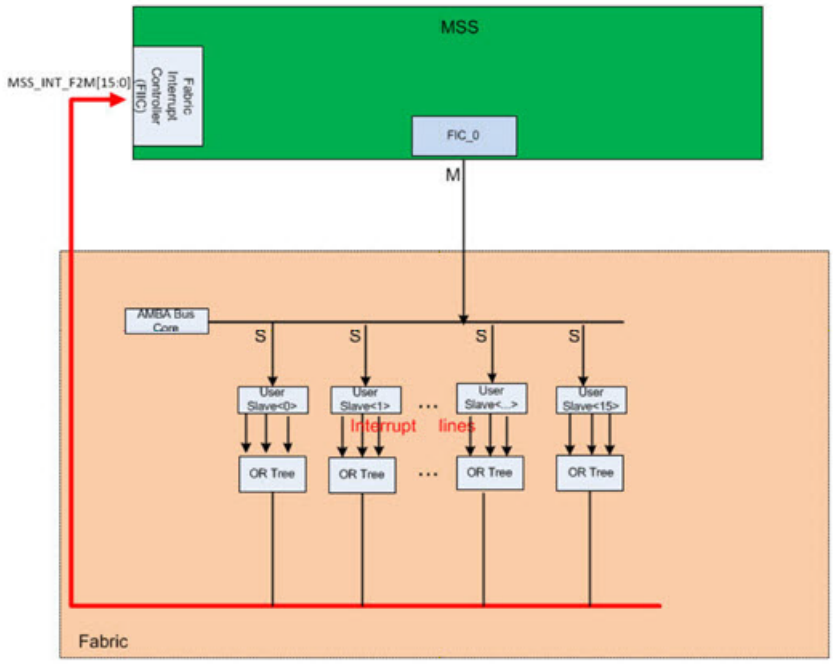

***Figure 2-23* • 16 Slaves with Dedicated Interrupt Access to MSS**

When there are more than 16 Slaves with interrupts, System Builder automatically instantiates the CoreInterrupt soft IP Core - a soft APB Interrupt Controller in the Fabric. Slave Interrupts from Slave 16 and above are connected to the 32 inputs (irqSource) of CoreInterrupt soft IP Core via cascaded OR-gates. The output of the CoreInterrupt, IRQ, is connected to MSS_INT_F2M[15]. For details about the CoreInterrupt core, refer to the CoreInterrupt Handbook.

Figure 2-24 shows 18 fabric slaves with interrupts. The interrupts from each of the 15 slaves are ORed in a cascaded OR-tree and connected to MSS_INT_F2M[14:0]. Bit 15 of MSS_INT_F2M is connected to the IRQ output of CoreInterrupt. Slave 0 through 14 have dedicated interrupt access to the MSS. Slave15, 16 and 17 have interrupt access to the MSS via the CoreInterrupt.

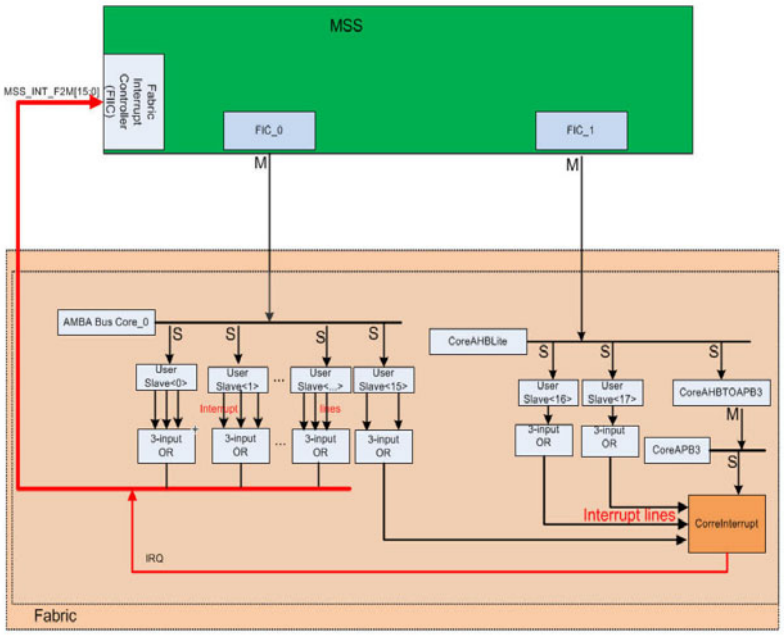

Figure 2-24 • **CoreInterrupt and Slave Interrupt Lines**

When the design has more than 47 slaves with interrupts, System Builder instantiates a second instance of CoreInterrupt to handle another 32 Slaves with Interrupts. The 32 inputs (irqSource) of the second CoreInterrupt are connected to interrupt lines from Slave 48 through 79 (via cascaded OR-gates). The IRQ output is connected to MSS_INT_F2M[14].

When the MSS Cortex-M3 processor receives an Interrupt from the Fabric, it can query the CoreInterrupt to determine from which Fabric Slave the Interrupt originates.

Fabric Slaves with Fixed Number of Interrupts

The Peripherals page of System Builder has a list of fabric slaves for you to build your subsystems:

- CoreAHBLSRAM
- CoreI2C
- CoreSPI
- CoreGPIO
- CoreTimer
- CoreUARTapb
- CorePWM

Figure 2-25 shows fabric slaves with a fixed number of interrupts.

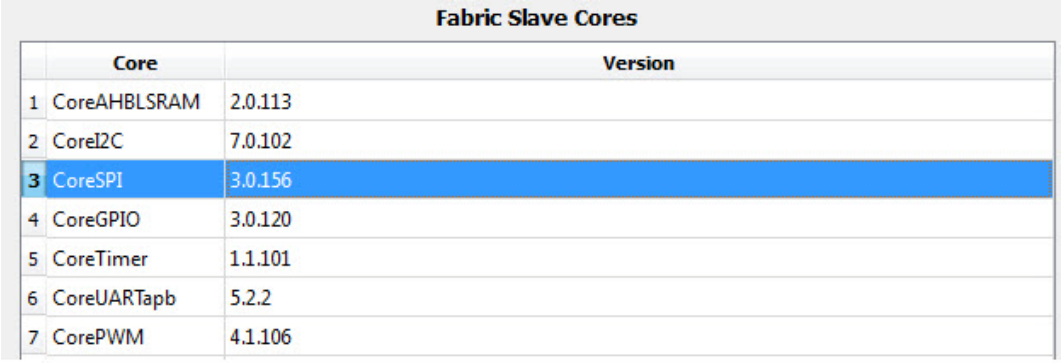

Figure 2-25 • **Fabric Slaves with Fixed Number of Interrupts**

For these fabric slaves, you do not need to configure the number of interrupt signals. They have a fixed number of interrupts. System Builder automatically connects the interrupts from these fabric slaves to the MSS_INT_F2M pins of MSS (or irqSource pins of CoreInterrupt) via cascaded OR gates.

Fabric Slaves with Configurable Number of Interrupts

When you want to add your own AMBA (AHBLite/APB3/AXI) compliant slave peripheral logic OR when you want to add other slave peripherals from the Libero Catalog window that are not listed under the 'Fabric Slave Cores' list to a subsystem with Cortex-M3(MSS) as the master, you have to add 'Fabric AMBA Slave' core from under the 'Fabric Slave Cores' list to the FIC_0/1 MSS Master Subsystems under the Subsystems list.

The number of Interrupts per fabric slave, up to a maximum of 8, is configurable. To specify the number of interrupts per slave, right-click the AMBA Slave added to a subsystem in the Peripherals page of System Builder to open its configurator.

Figure 2-26 shows the AMBA Fabric Slave in the Peripherals page.

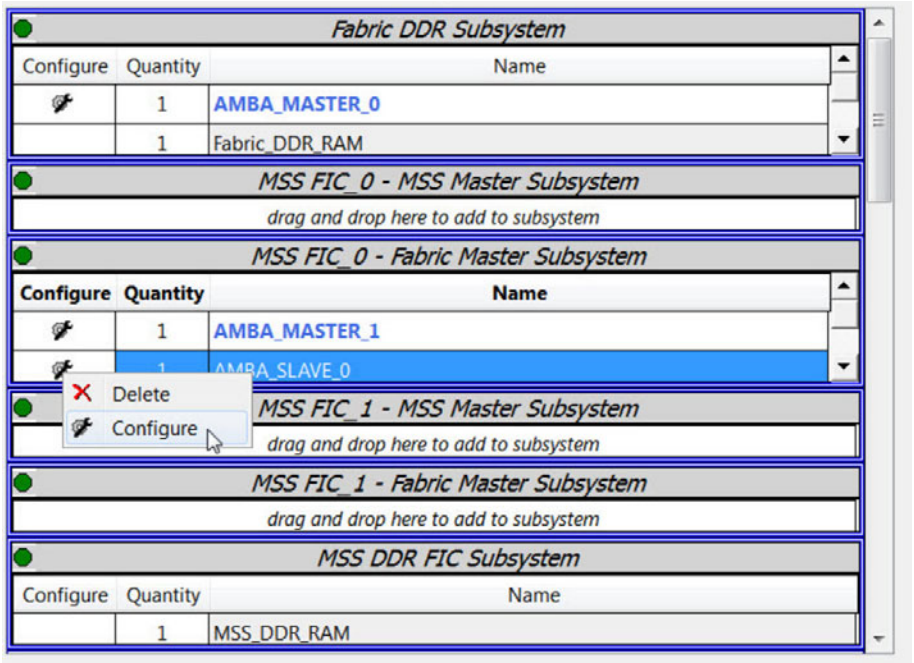

Figure 2-26 • **AMBA Fabric Slave in Peripherals Page**

In the Configurator, specify the Interface Type (AHBLite/APB3/AXI) and the number of Interrupts you need (Maximum 8). See Figure 2-27.

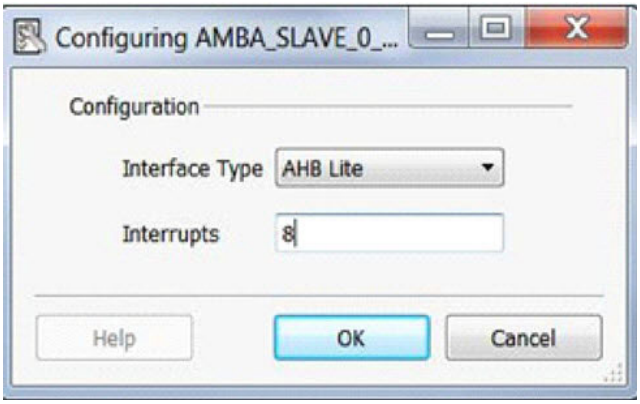

Figure 2-27 • **Number of Interrupts Per Slave**

For fabric slaves with a configurable number of interrupts, System Builder exposes to the top level System Builder block the OR tree inputs corresponding to each fabric slave interrupts. You need to connect these top level input ports to the interrupt trigger signals of the slave peripheral logic outside the System Builder block.

Interrupts Page and Pin Assignment

By default, System Builder assigns pins of MSS_INT_F2M[15:0] to specific slaves. You can change the pin assignment from System Builder's Interrupt page.

To assign/unassign a fabric slave instance to a specific MSS_INT_F2M pin, click the corresponding fabric slave instance name and choose an option from the drop-down menu.

Figure 2-28 shows MSS interrupt signals in the Interrupts page.

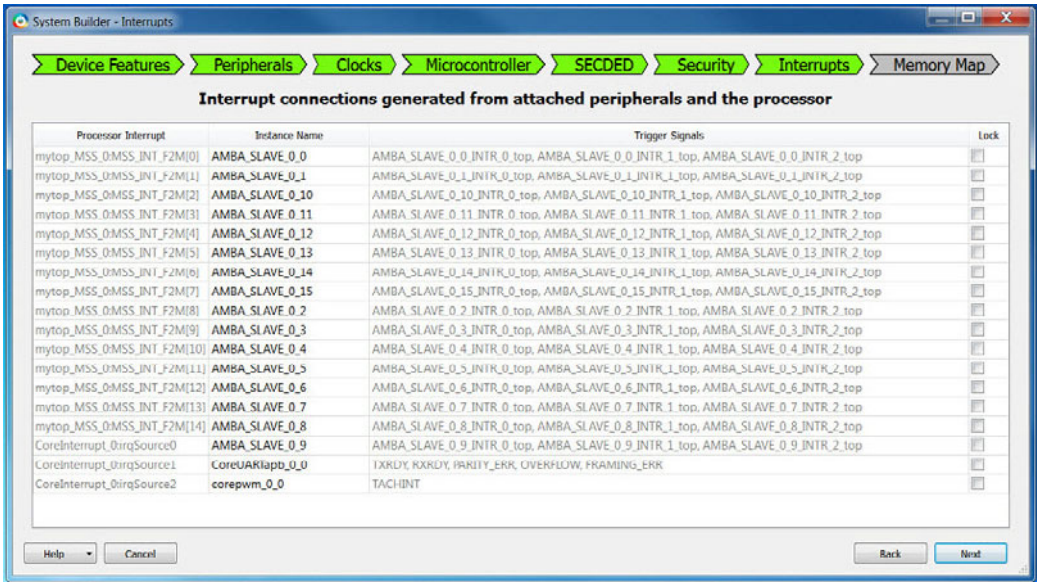

Figure 2-28 • **MSS Interrupt Signals**

This page also enables you to lock the interrupt pin assignment of a fabric slave core's interrupt pins to a desired MSS_INT_F2M[n] pin. A maximum of 16 different fabric slave cores' interrupt pins can be directly connected to the MSS_INT_F2M[15:0]. To lock the pin assignment, check the Lock checkbox at the far right.

When a specific MSS_INT_F2M[n] pin or irqSource<n> pin is assigned to a specific slave, the "locked" assignment is honored when you regenerate your design in System Builder or add/remove slaves with interrupts incrementally. The pin assignment of the specific slave interrupt line to MSS_INT_F2M[n] remains unchanged after System Builder regenerates your system.

You do not need to lock the interrupt lines if either of the following is true:

- You are satisfied with the default interrupt pin assignment
- You do not want to modify or regenerate your design in System Builder (with or without changes to the slave peripherals)

If you choose not to "lock" the interrupt assignment for a slave the first time you generate the System Builder block and then after generation decide to modify the design (adding or deleting slave peripherals), the interrupt assignment for the slave may change on regeneration.

The CoreInterrupt is an APB3 slave. The Memory Map of the CoreInterrupt core from the MSS perspective displays under the CoreAPB3 bus in the Memory Map page of System Builder (Figure 2-29). The memory address of the CoreInterrupt is design-specific. Your design may show a memory address different from this figure.

Memory Map Page

This page displays the memory map for each of your subsystems. You can make limited modifications to the memory map within a subsystem. "The Select Bus to View or Assign Peripheral(s)" panel displays one bus interface for each subsystem in your design. These buses are internally generated (they are part of the System Builder block).

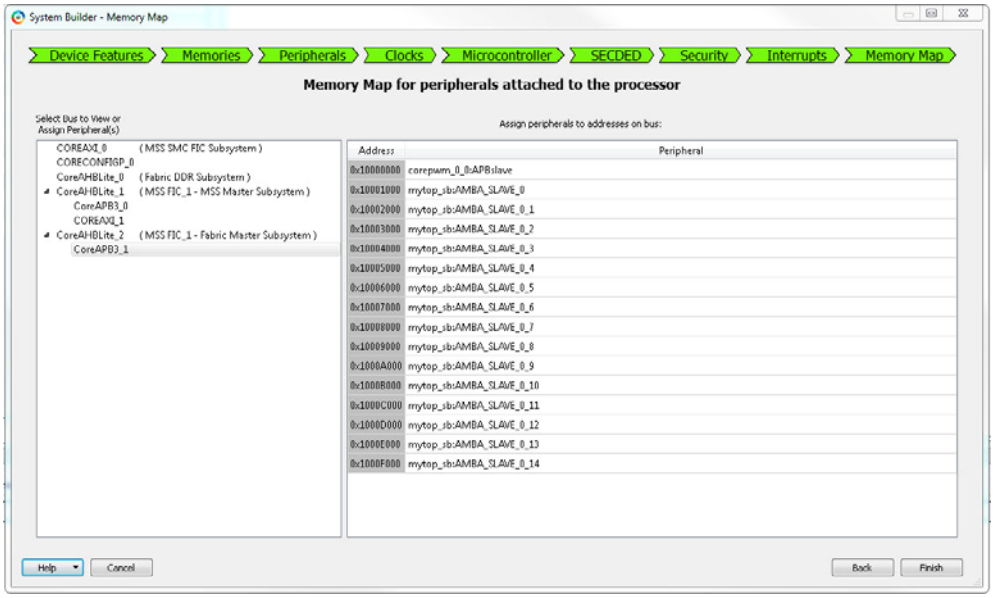

Figure 2-29 • **Memory Map of Subsytems with different slave types**

If there are multiple slaves on a bus, you can use the drop-down lists in the "Assign peripherals to addresses on the bus" panel to modify slave addresses.

Note: For the MSS FIC0/1 Fabric Master subsystems, address ranges 0x00000000-0x0FFFFFFF, 0x20000000-0x2FFFFFFF, 0x40000000-0x4FFFFFFF, and 0x60000000-0x6FFFFFFF all map into the MSS.

Refer to the SmartFusion2 Fabric User's Guide for details.

For the MSS master (Cortex-M3) to access the fabric space via FIC_0/1, or for the fabric masters to access fabric slaves and the MSS space via FIC_0/1, or for the fabric masters to access external DDR memory via MDDR/FDDR and for other possible use cases, you should know the bus core configuration and the memory map of different slaves in the subsystem. The System Builder Memory Map page is the place to view the memory map of different subsystems in use and the base addresses of different slaves.

An example of each of the Bus core configuration for MSS FIC_0 - Fabric Master Subsystem and MSS FIC_1 - Fabric Master Subsystem is shown below.

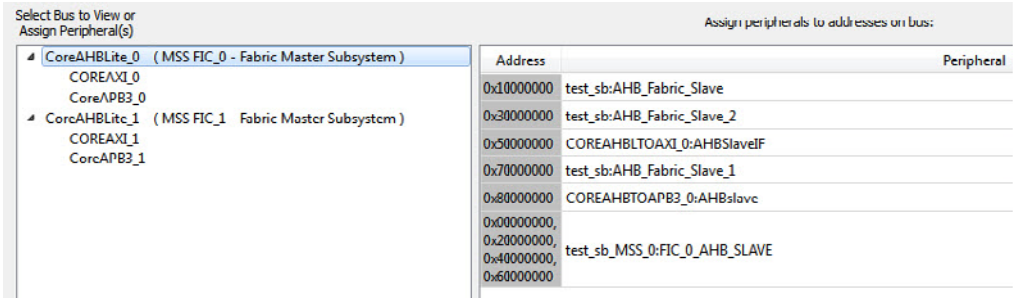

***Figure 2-30* • Memory Map of MSS FIC_0 - Fabric Master Subsystem**

.

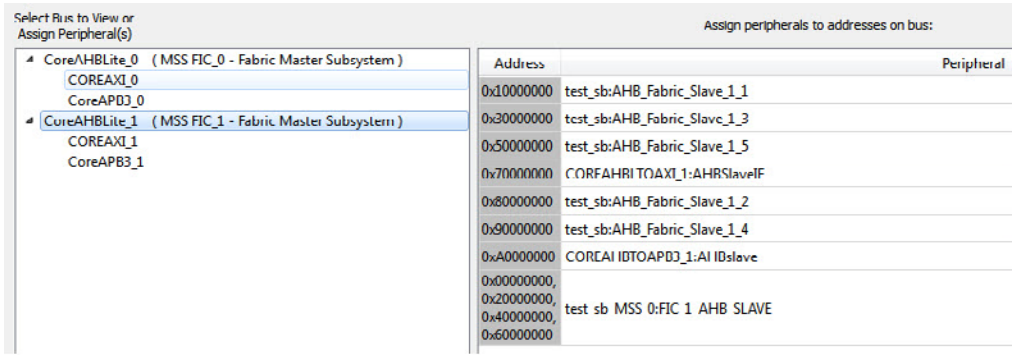

***Figure 2-31* • Memory Map of MSS FIC_1 - Fabric Master Subsystem**

As can be seen in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAHBLite - Total 4GB space apportioned into 16 slave slots of 256MB each
  - MSS (FIC_0/1_AHB_SLAVE) is connected as an AHBLite slave to the combined region slave slot of CoreAHBLite
  - Other AHBLite fabric slaves are connected to the normal slave slots of CoreAHBLite
- CoreAXI - Total 16MB space apportioned into 16 slave slots of 1MB each
  - AXI fabric slaves are connected to the slave slots of CoreAXI
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

System Builder Subsystems

A subsystem is defined as at least one AMBA master and at least one AMBA slave attached to an AMBA bus accessible in the MSS memory space through the Fabric Interface Controllers (FIC_0/1) or the DDR/SMC_FIC Controller. The AMBA master can be a fabric master or the MSS.

MSS FIC_0 - MSS Master Subsystem

This subsystem is available by default. It enables an MSS Master (Cortex-M3) to access the fabric space through the Fabric Interface Controller FIC_0 (Table 2-1).

If you want the Cortex-M3 to access a slave in the fabric, drag and drop it into this subsystem. You can drag and drop fabric slaves of different interface types (AHBLite, AXI or APB) into this subsystem.

For details, refer to "Use Case #3" FIC0/1_MSS _Master_Subystems with different types of slaves (AXI/AHBLite/APB3) in Appendix A, "Use Cases".

Figure 2-32 shows the MSS FIC_0 - MSS Master Subsystem.

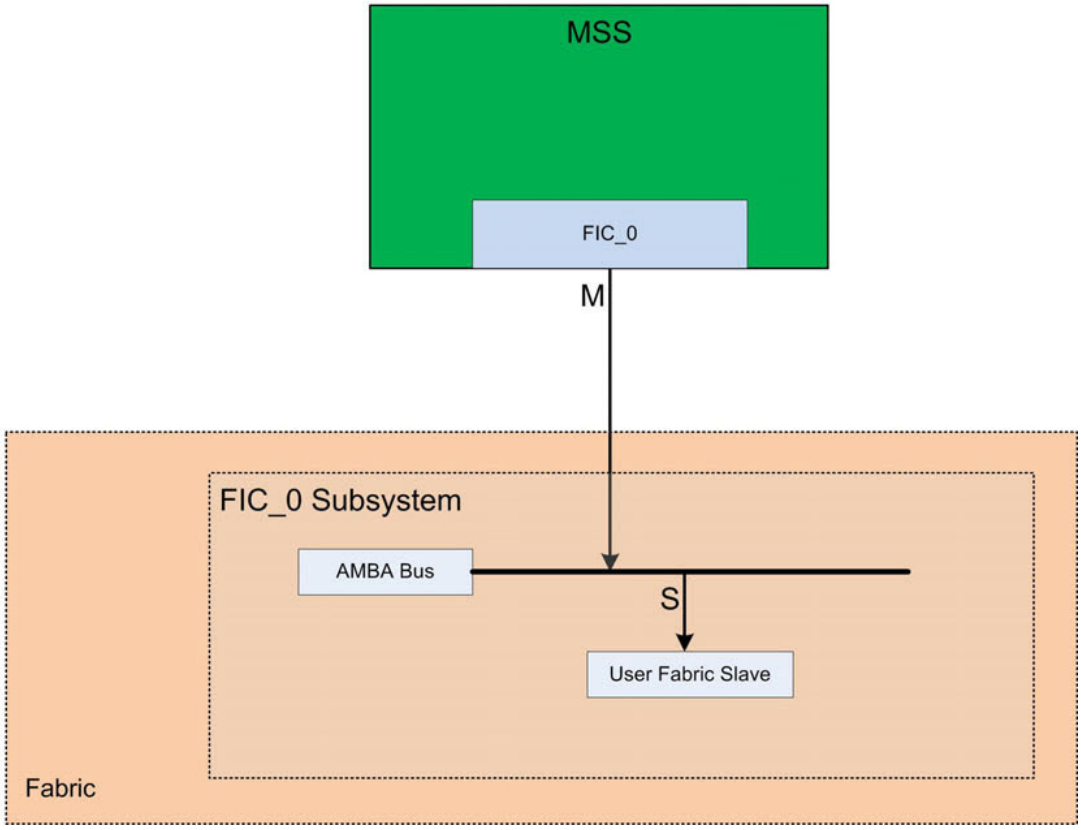

***Figure 2-32* • MSS FIC_0 - MSS Master Subsystem**

***Table 2-1* • MSS FIC_0 - MSS Master Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
System Builder Master BIF (AHBLite/APB3 BIF)	Slave	Connect to User Fabric Slave's AMBA Slave BIF. For each User Fabric Slave, connect its Slave BIF to the corresponding Slave BIF on the System Builder block.
FIC_0_CLK	Out	FIC_0 subsystem clock. Connect to User Fabric Slaves.
FIC_0_LOCK	Out	Asserts when FIC_0_CLK is valid. If your User Fabric Master/ slave has a PLL LOCK input, connect it to this pin.

MSS FIC_0 - Fabric Master Subsystem

This subsystem is available by default. It enables a Fabric Master to access the MSS memory space through FIC_0 in addition to other Fabric peripherals (Table 2-2). If you are using this subsystem, you must drag and drop a Fabric AMBA Master into this subsystem and then connect your actual AMBA Master peripheral in the top level Smart Design. This subsystem supports up to four AHBLite/AXI fabric masters, and up to one APB3 fabric master. However, you cannot mix and use more than one master type

When the masters in the subsystem are all AHBLite or all AXI, you may add a mix of different user slave interface types (Fabric AMBA Slaves - AXI/AHBLite/APB3) to this subsystem.

For details, refer to "Use Case #1" FIC0/1_Fabric_Master_Subsystems with one/more AHBLite fabric masters and different types of slaves (AXI/AHBLite/APB3) and "Use Case #2" FIC0/1_Fabric_Master_Subsystems with one/more AXI fabric masters and different types of slaves (AXI/AHBLite/APB3) in Appendix A, "Use Cases".

Figure 2-33 shows the MSS FIC_0 - Fabric Master Subsystem.

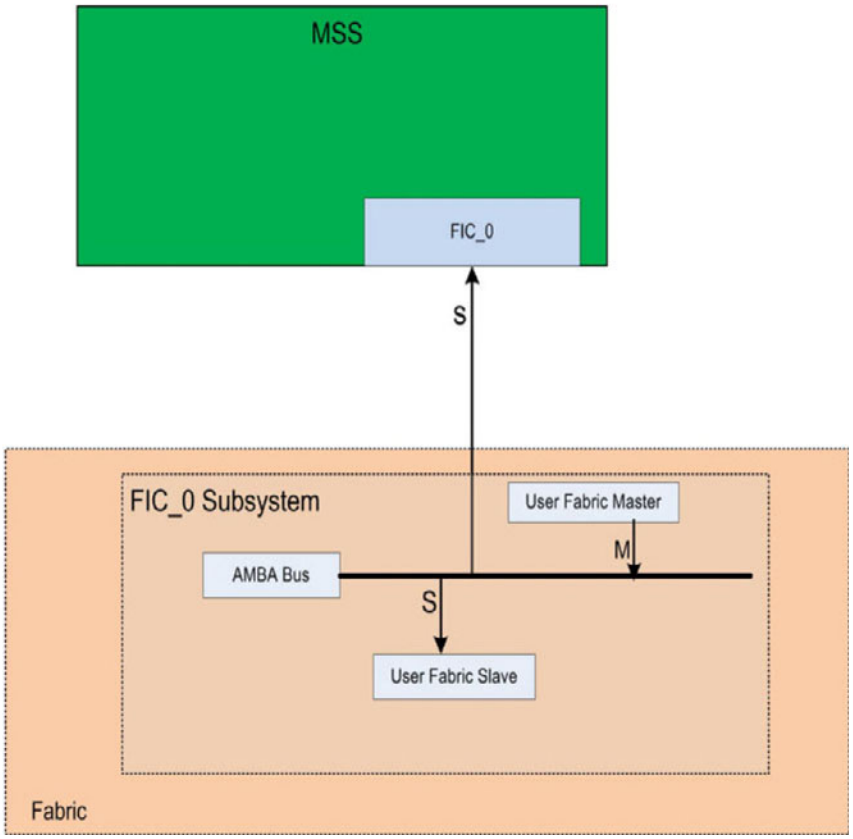

***Figure 2-33* • MSS FIC_0 - Fabric Master Subsystem**

***Table 2-2* • MSS FIC_0 Fabric Master Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
System Builder Master BIF (AHBLite/AXI/APB3 BIF)	Master	Connect to User Fabric Master's AHBLite/AXI/APB3 Master BIF. Up to four AHBLite/AXI Masters and one APB3 Master BIF are supported.
System Builder Slave BIF	Slave	Connect to User Fabric Slave's AMBA Slave BIF. For each User Fabric Slave, connect its Slave BIF to the corresponding Slave BIF on the System Builder block.
FIC_0_CLK	Out	FIC_0 subsystem clock. Connect to User Fabric Master.
FIC_0_LOCK	Out	Asserts when FIC_0_CLK is valid. If your User Fabric Master/ slave has a PLL LOCK input, connect it to this pin.

MSS FIC_1 - MSS Master Subsystem - M2S050 and Larger Devices

This subsystem is available by default. It enables an MSS Master (Cortex-M3) to access the fabric space through FIC_1 (Table 2-3). If you want the Cortex-M3 to access a slave in the fabric, drag and drop it into this subsystem. You can drag and drop fabric slaves of different interface types (AHBLite, AXI or APB) into this subsystem.

You can add a mix of different AMBA interface slaves (Fabric AMBA Slaves - AXI/AHBLite/APB3) to this subsystem.

For details, refer to "Use Case #3" FIC0/1_MSS _Master_Subystems with different types of slaves (AXI/AHBLite/APB3) in Appendix A, "Use Cases".

Figure 2-34 shows the MSS FIC_1 - MSS Master Subsystem.

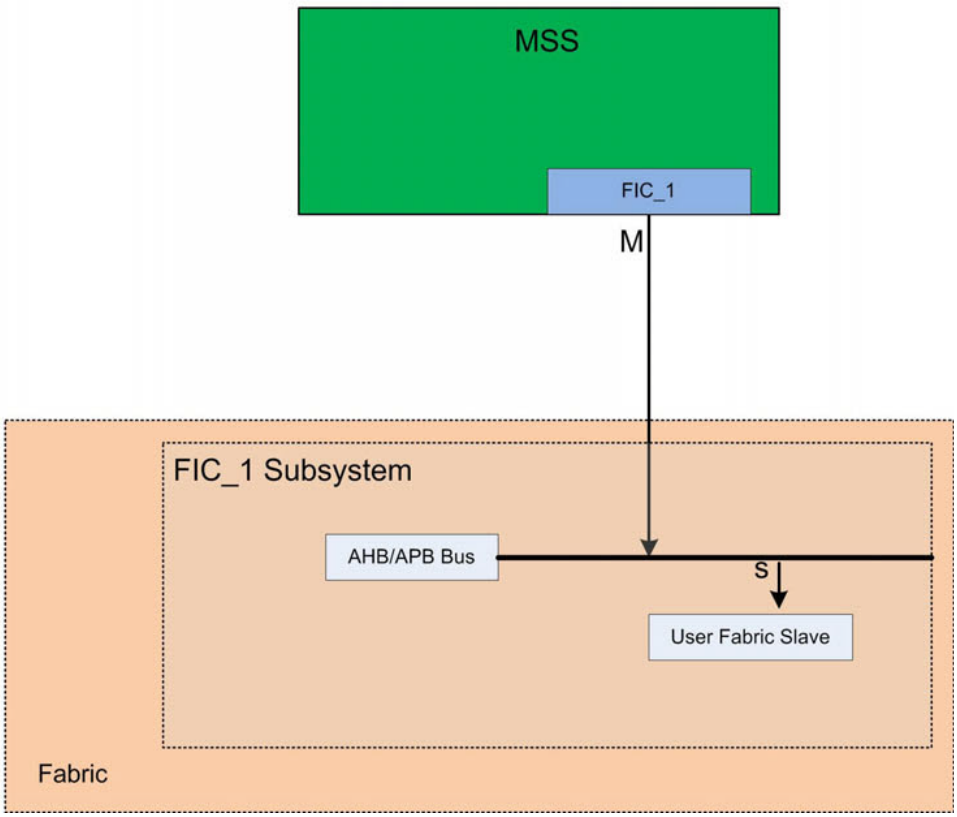

***Figure 2-34* • MSS FIC_1 - MSS Master Subsystem**

***Table 2-3* • MSS FIC_1 - MSS Master Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
System Builder Slave BIF	Slave	Connect to User Fabric Slave's AMBA Slave BIF. For each User Fabric Slave, connect its Slave BIF to the corresponding Slave BIF on the System Builder block.
FIC_1_CLK	Out	FIC_1 subsystem clock. Connect to User Fabric Slaves.
FIC_1_LOCK	Out	Asserts when FIC_1_CLK is valid. If your User Fabric Master/ CoreSysServices has a PLL LOCK input, connect it to this pin.

MSS FIC_1 - Fabric Master Subsystem - M2S050 and Larger Devices Only

This subsystem is available by default. It enables a Fabric Master to access the MSS memory space through FIC_1 in addition to other Fabric peripherals (Table 2-4). If you are using this subsystem, you must drag a Fabric AMBA Master into this subsystem and then connect your AMBA peripheral in the top level Smart Design. This subsystem supports up to four AHBLite/AXI fabric masters, and up to one APB3 fabric master. However, you cannot mix and use more than one master type.

When the masters in the subsystem are all AHBLite or all AXI, you may add a mix of different user slave types (Fabric AMBA Slaves - AXI/AHBLite/APB3).

For details, refer to "Use Case #1" FIC0/1_Fabric_Master_Subystems with one/more AHBLite fabric masters and different types of slaves (AXI/AHBLite/APB3) and "Use Case #2" FIC0/1_Fabric_Master_Subystems with one/more AXI fabric masters and different types of slaves (AXI/AHBLite/APB3) in Appendix A, "Use Cases".

Figure 2-35 shows the MSS FIC 1 - Fabric Master Subsystem.

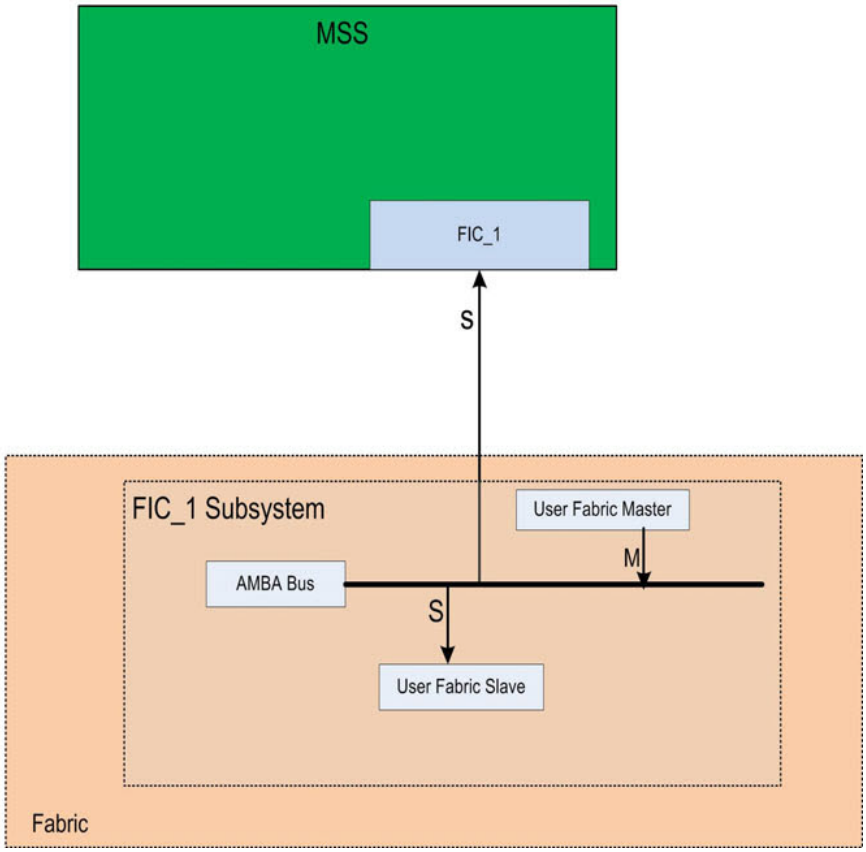

Figure 2-35 • **MSS FIC_1 - Fabric Master Subsystem**

Table 2-4 • **MSS FIC 1 - Fabric Master Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
System Builder Master BIF (AHBLite/AXI/APB3 BIF)	Master	Connect to User Fabric Master's AHBLite/AXI/APB3 Master BIF. Up to four AHBLite/AXI Masters and 1 APB Master BIF are supported.
System Builder Slave BIF	Slave	Connect to User Fabric Slave's AMBA Slave BIF. For each User Fabric Slave, connect its Slave BIF to the corresponding Slave BIF on the System.
FIC_1_CLK	Out	FIC_1 subsystem clock. Connect to User Fabric Master.
FIC_1_LOCK	Out	Asserts when FIC_1_CLK is valid. If your User Fabric Master/ CoreSysServices has a PLL LOCK input, connect it to this pin.

MSS DDR FIC Subsystem

This subsystem is available if you check the **MSS External Memory** checkbox and select the **MDDR** radio button on the Device Features page (Table 2-5).

When available, the MSS_DDR_RAM peripheral displays automatically. The Cortex-M3 is a default master on this subsystem.

Optionally, you may choose to have a Fabric AXI/AHBLite Master access the external DDR memory via MDDR by instantiating a Fabric AMBA Master (AXI/AHBLite) core into this subsystem (drag Fabric AMBA Master and drop it in the MSS DDR FIC Subsystem to instantiate). Refer to the MDDR Configuration Guide for more information. Adding an APB3 Master to this subsystem is not supported.

Optionally, you may add Fabric Slaves to this subsystem. Adding a mix of user slave types (Fabric AMBA Slaves - AXI/AHBLite/APB3) to this subsystem is supported.

For details, refer to Appendix A, "Use Cases":

- "Use Case #4" MSS DDR_FIC_Subsystem with one AHBLite fabric master and different types of slaves (AXI/AHBLite/APB3)
- "Use Case #5" MSS DDR_FIC_Subsystem with one AXI fabric master and different types of slaves (AXI/AHBLite/APB3) and
- "Use Case #6" MSS DDR_FIC_Subsystem with one AXI fabric master and no other slaves

Figure 2-36 shows the MDDR Fabric Master Subsystem.

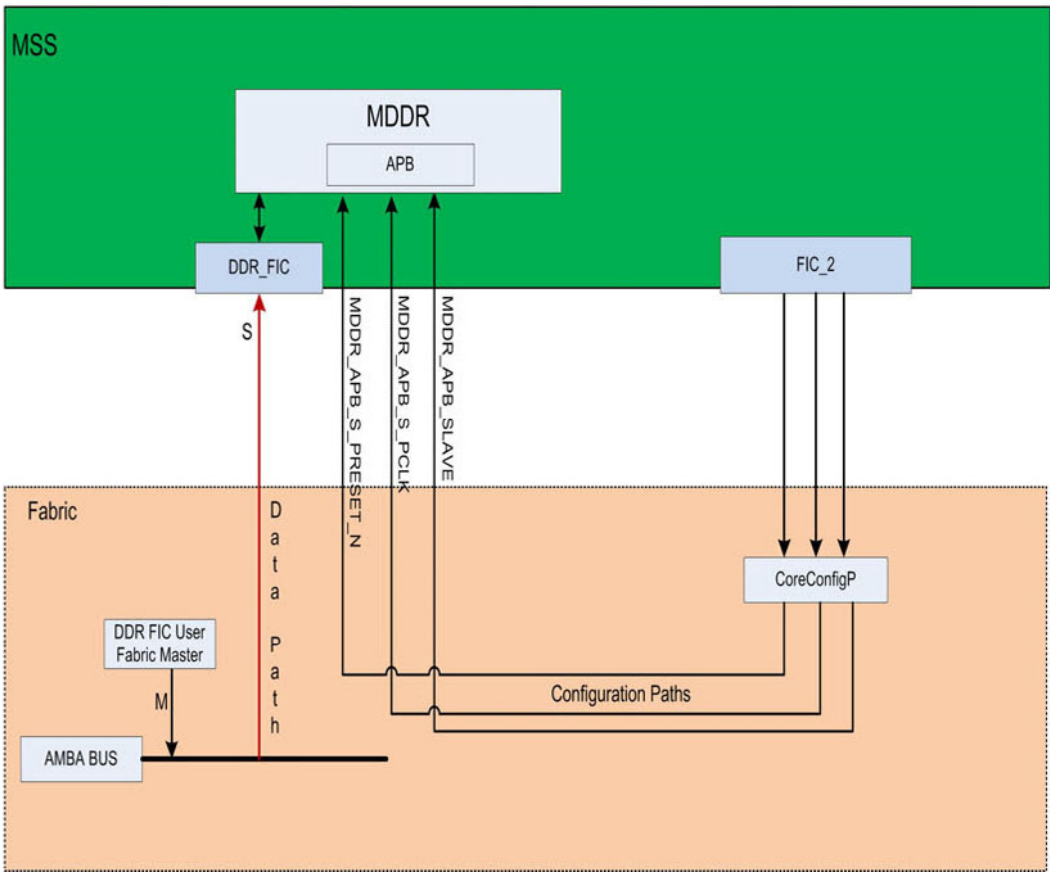

***Figure 2-36* • MDDR Fabric Master Subsystem**

The DDR FIC and the SMC FIC Subsystems are mutually exclusive. You can have one or the other but you cannot enable both. Table 2-5 shows the port list.

***Table 2-5* • MSS DDR FIC Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
System Builder Master (AHBLite/AXI) BIF	Master	Connect to User Fabric Master's AMBA Master BIF.
System Builder Slave (AHBLite/AXI/APB3) BIF	Slave	Connect to User Fabric Master's AMBA Slave BIF.
MSS_DDR_FIC_SUBSYSTEM_PINS: MSS_DDR_FIC_SUBSYSTEM_CLK	Out	MSS DDR subsystem clock. Connect to User Fabric Master and User Fabric Slaves in this subsystem.
MSS_DDR_FIC_SUBSYSTEM_PINS: MSS_DDR_FIC_SUBSYSTEM_LOCK	Out	Asserts when MSS_DDR_FIC_SUBSYSTEM_CLK is valid. If your User Fabric Master/User Fabric Slaves have PLL LOCK inputs, connect them to this pin.
MSS_DDR_FIC_SUBSYSTEM_PINS: MDDR_AXI_S_RMW	In	Indicates whether all bytes of a 64 bit lane are valid for all beats of an AXI transfer. 0: Indicates that all bytes in all beats are valid in the burst and the controller should default to write commands. 1: Indicates that some bytes are invalid and the controller should default to RMW commands. This is classed as an AXI write address channel sideband signal and is valid with the AWVALID signal. Only used when ECC is enabled. Tie low if not used.
MDDR_PADS	In/Out	DDR Memory Physical Interface PADs. Refer to the SmartFusion2 DDR Controller Configuration User's Guide for details.

MSS SMC FIC Subsystem

This subsystem is available if you check the **MSS External Memory** checkbox and select the **Soft Memory Controller (SMC)** radio button on the Device Features page. In this configuration, the MSS accesses external Single Data Rate DRAM or Asynchronous memories via a soft memory controller instantiated in the FPGA fabric, such as CoreSDR_AXI and CoreSDR_AHB. The Cortex-M3 is a default master on the MSS SMC FIC subsystem.

Figure 2-37 shows the SMC FIC Subsystem.

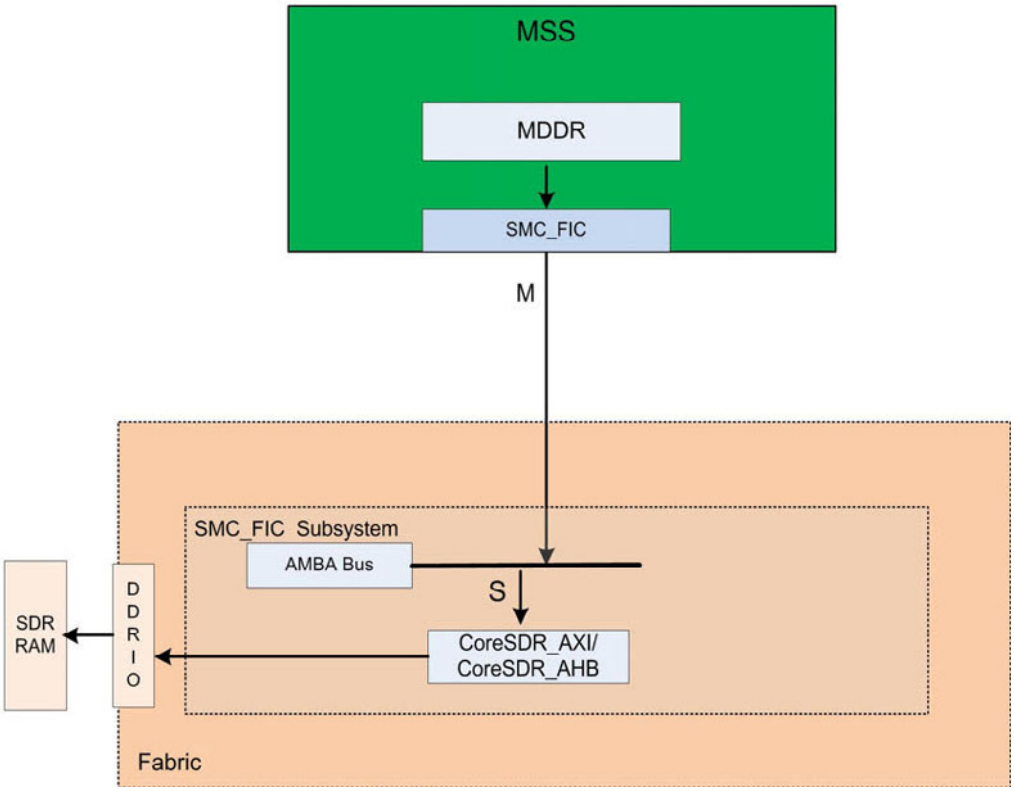

Figure 2-37 • **SMC FIC Subsystem**

The DDR FIC and the SMC FIC Subsystems are mutually exclusive. You can have one or the other but you cannot enable both. Table 2-6 shows the port list.

Table 2-6 • **SMC FIC Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
MSS_SMC_0_PINS	Pin Group	Connect to non-DDR based External Memory.

Fabric DDR Subsystem

This subsystem is available if you check the Fabric External DDR Memory (**FDDR**) checkbox on the Device Features page. When available, the Fabric_DDR_RAM peripheral is automatically shown in this subsystem. You must provide a master for the Fabric_DDR_RAM. This can be done in two ways:

- If you intend to have a Fabric Master access the Fabric DDR, then you click and drag a Fabric AMBA Master core into this subsystem from the available cores list and configure it as either AHBLite or AXI. Only one Master is supported on this subsystem. Figure 2-38 shows the FDDR Subsystem with a Fabric Master. When you generate your system, you get a BIF where you must connect your actual Fabric Master IP in the top level SmartDesign. Optionally, you may add AMBA Slaves to this subsystem. Adding a mix of user slave types (Fabric AMBA Slaves - AXI/AHBLite/APB3) to this subsystem is supported.
- If you want to access the Fabric DDR RAM from a FIC_0/1 subsystem, you can also click and drag the Fabric_DDR_RAM peripheral into one of the MSS FIC_0/1 - MSS Master Subsystems OR MSS FIC_0/1 - Fabric Master Subsystems. This action eliminates the Fabric DDR Subsystem and makes the Fabric DDR Controller part of the subsystem you dragged it into. The Master of the Fabric DDR will now be the same as the master of the subsystem it is in (i.e., Cortex-M3 in the MSS or a Fabric Master). For example, If you intend to

have the Cortex-M3 access the Fabric DDR, click and drag the Fabric_DDR_RAM peripheral into one of the MSS FIC_0/1 - MSS Master Subsystems. If you do this, on the Clocks page, you must set the FDDR_SUBSYSTEM_CLK frequency to be equal to the frequency of FIC_0/1_CLK(depending on whether you dragged the Fabric_DDR_RAM into a FIC_0 or FIC_1 subsystem).

- You can also click and drag the Fabric_DDR_RAM peripheral into the MSS DDR FIC Subsystem. This eliminates the Fabric DDR Subsystem and makes the Fabric DDR Controller part of the MSS DDR FIC subsystem. The master of the Fabric DDR will now be the same as the master of the MSS_DDR. When you generate your system, you get a BIF where you must connect your actual Fabric Master IP in the top level SmartDesign. Also, on the Clocks page, you must set the FDDR_SUBSYSTEM_CLK frequency to be equal to the frequency of DDR/SMC_FIC_CLK.

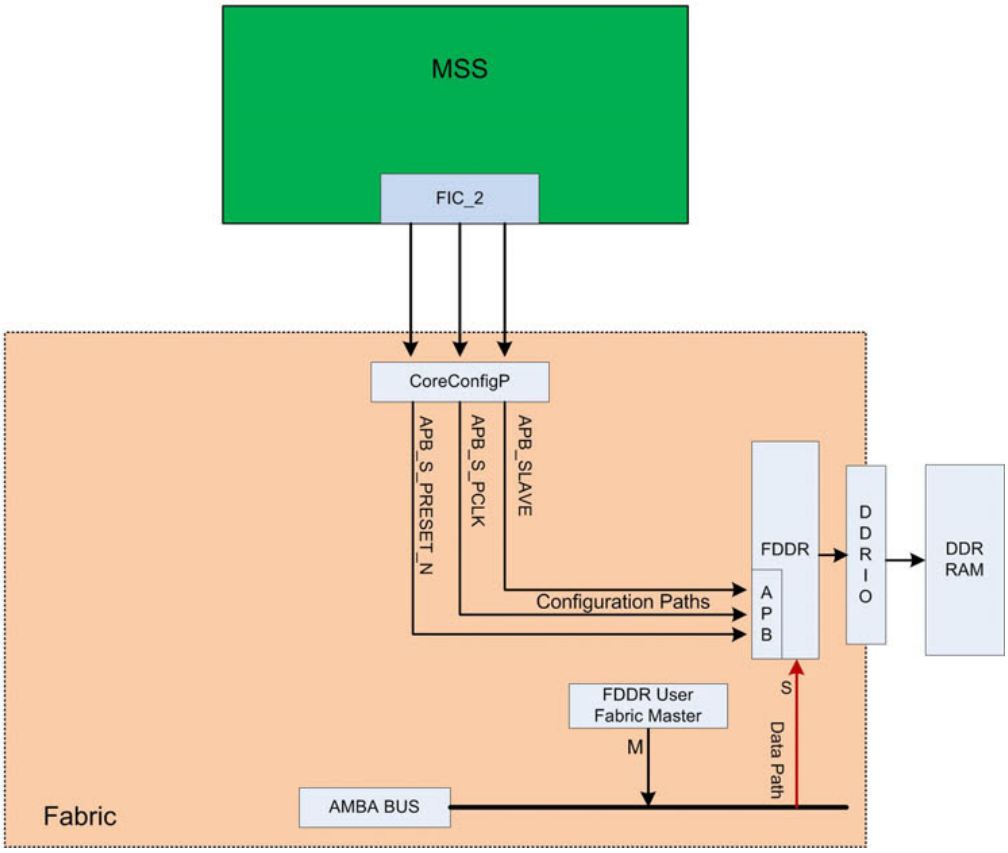

***Figure 2-38* • FDDR Subsystem with a Fabric Master**

For details about this subsystem, refer to Appendix A, "Use Cases":

- "Use Case #7" Fabric_DDR_Subsystem with one AHBLite fabric master and different types of slaves (AXI/AHBLite/APB3)
- "Use Case #8" Fabric_DDR_Subsystem with one AXI fabric master and different types of slaves (AXI/AHBLite/APB3)
- "Use Case #9" Fabric_DDR_Subsystem with one AXI fabric master and no other slaves

Table 2-7 lists the Fabric DDR Subsystem Ports.

Table 2-7 • **Fabric DDR Subsystem Port List**

Pin/BIF	Direction	Functionality and Usage
System Builder Master (AHBLite /AXI) BIF	Master	Connect to User Fabric Master's AMBA Master BIF.
System Builder Slave (AHBLite/AXI/APB3) BIF	Slave	Connect to User Fabric Master's AMBA Slave BIF.
FDDR_SUBSYSTEM_PINS: FDDR_SUBSYSTEM_CLK	In	Fabric DDR subsystem clock. You must provide a clock for your Fabric DDR Subsystem using one of the following options: Instantiate Fabric CCC in SmartDesign to generate the clock for the FDDR subsystem. The Fabric CCC generates a clock and PLL lock signals. Connect Fabric CCC's Clock output (GL0/1/2/3) to: • FDDR_SUBSYSTEM_CLK • User Fabric Master's Clock Input • User Fabric Slaves' Clock Inputs If you want to use the same clock frequency as one of the other subsystems (FIC_0/1 or MSS DDR), instead of instantiating a Fabric CCC, you can get the appropriate clock and lock signals from your chosen subsystem's pins. Connect your chosen clock signal (one of FIC_0_CLK, FIC_1_CLK, or MSS_DDR_FIC_SUBSYSTEM_CLK) to: • FDDR_SUBSYSTEM_CLK • User Fabric Master's Clock Input • User Fabric Slaves' Clock Inputs
FDDR_SUBSYSTEM_PINS: FDDR_SUBSYSTEM_CLK_PLL_LOCK	In	Asserts when FDDR_SUBSYSTEM_CLK is valid. Depending on your selection of the driver of FDDR_SUBSYSTEM_CLK, connect to one of the following: • Fabric CCC:LOCK • System Builder block's FIC_0_LOCK • FIC_1_LOCK • MSS_DDR_FIC_SUBSYSTEM_LOCK
FDDR_SUBSYSTEM_PIN: FDDR_AXI_S_RMW	In	Indicates whether all bytes of a 64 bit lane are valid for all beats of an AXI transfer. 0: Indicates that all bytes in all beats are valid in the burst and the controller should default to write commands 1: Indicates that some bytes are invalid and the controller should default to RMW commands. This is classed as an AXI write address channel sideband signal and is valid with the AWVALID signal. Only used when ECC is enabled. Tie low if not used.

***Table 2-7* • Fabric DDR Subsystem Port List (continued)**

Pin/BIF	Direction	Functionality and Usage
FDDR_SUBSYSTEM_PINS: FDDR_ SUBSYSTEM_RESET_N	In	Reset the FDDR subsystems. You may connect the MSS_READY signal or DDR_READY signal or other user fabric logic signal to this pin at the top level.
FDDR_PADS	In/Out	DDR Memory Physical Interface PADs. Refer to the SmartFusion2 DDR Controller Configuration User's Guide for details.

SERDESIF<0/1/2/3> Subsystems and SERDES Configuration Path

This subsystem is available if you check the SERDESIF_<0/1/2/3> checkbox in the Device Features page of System Builder.

Based on your selection in the Device Features page, System Builder automatically generates and exposes the SERDES configuration data bus interface (APB3 master BIF) and the clock and reset pins on the top level System Builder design.

Figure 2-39 shows the SERDESIF_<0/1/2/3> Subsystem.

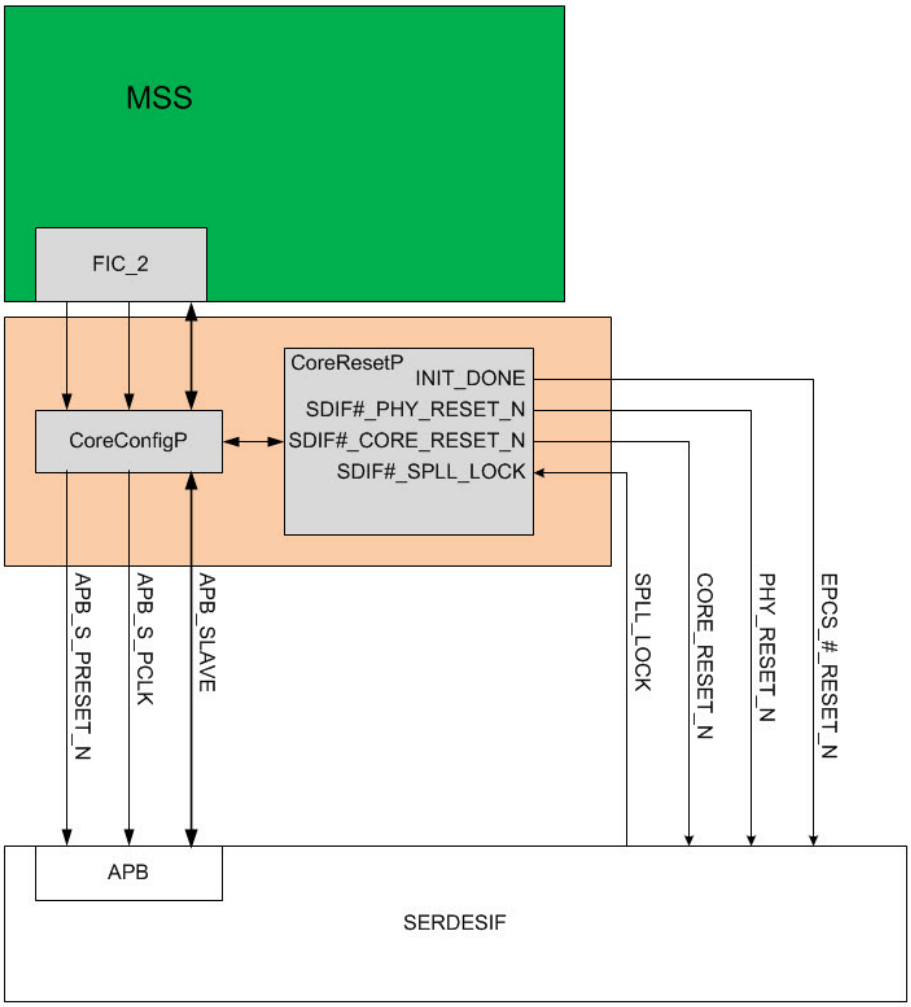

***Figure 2-39* • SERDESIF_<0/1/2/3> Subsystem**

You will have to instantiate the SERDES blocks in the SmartDesign and connect their clock reset and configuration pins and BIFs to System Builder as shown in Table 2-8.

The definitive name of a SERDES instance (SERDES_0/1/2/3) is visible in its configurator (Figure 2-40). Correlate this with the correct SDIF_<0/1/2/3>_ PINS (Table 2-8) on the System Builder block. For example, connect pins in the SDIF_0_PINS group to SERDESIF_0, SDIF_1_PINS to SERDES_1, etc.

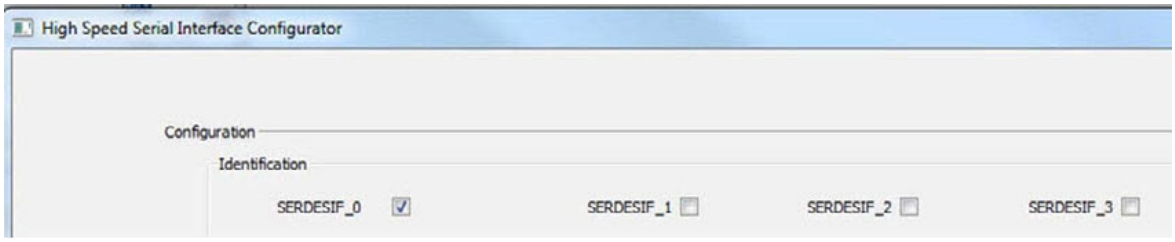

***Figure 2-40* • SERDES Instance Identification (Partial View)**

***Table 2-8* • SERDES Configuration Port List**

Pin	Direction	Functionality and Usage
SDIF(0-3)_INIT_APB BIF	Slave	For each SERDES block, System Builder generates an APB Slave BIF. These appear at the bottom of the System Builder block, and are called SDIF(0-3)_ INIT_APB. Connect the APB Slave BIF on each SERDES block to its corresponding SDIF(0-3)_APB_Slave BIF on the System Builder block.
INIT_PINS:INIT_APB_S_PCLK	Out	SERDES APB Configuration Path Clock. Connect to the INIT_APB_S_PCLK input port of all SERDES instances
INIT_PINS:INIT_APB_S_PRESET_N	Out	SERDES APB Configuration Path Reset. Connect to the INIT_APB_S_PRESET_N port of all SERDES instances
SDIF(0-3)_PINS: SDIF(0-3)_PERST_N	In	L2 and P2 are low power states for the Link and PHY interface in a PCI Express (PCIe) system. A power management component in a PCIe system controls the exit from the L2/P2 state. Part of the sequence when emerging from the low power state involves assertion and release of the PCI Express Reset (SDIF(0-3)_PERST_N pins). If you are not using PCIe, tie this pin high.
SDIF(0-3)_PINS: SDIF_(0-3)_PHY_RESET_N	Out	Deasserts to bring SERDES PHY interface out of reset. Connect to the PHY_RESET_N reset input of the corresponding SERDES instance when in PCIe or XAUI mode. For EPCS mode, tie this port HIGH.
SDIF(0-3)_PINS: SDIF_(0-3)_CORE_RESET_N	Out	Deasserts to bring SERDES PCIe and XAUI Core Logic out of reset. Connect to the CORE_RESET_N reset input of the corresponding SERDES instance when in PCIe or XAUI modes. For EPCS mode, tie this port HIGH.

***Table 2-8* • SERDES Configuration Port List (continued)**

Pin	Direction	Functionality and Usage
SDIF(0-3)_PINS: SDIF_(0-3)_SPLL_LOCK	In	Asserts when the SERDES instance's interface PLL has locked. Connect to the SPLL_LOCK output when using the PCIe and XAUI modes of the SERDES. For EPCS mode, tie this port HIGH.
INIT_DONE	OUT	Asserts high when all of the peripherals have been provisioned by the MSS. The SERDES should be held in reset until this port asserts. This port should be connected to the SERDES in the following manner: • PCIe mode: No connection to the SERDES. This same function is handled by PHY_RESET_N. • XAUI mode: No connection to the SERDES. This same function is handled by PHY_RESET_N. • EPCS mode: Connect to EPCS_#_RESET_N.

MSS Peripherals Subsystem

This subsystem is available by default. It contains the list of hard peripherals inside the MSS. You can choose to enable/disable/configure each of these peripherals in the System Builder Peripherals page. The hard peripherals include GPIO, MMUART, I2C, SPI, USB, CAN and Ethernet MAC. When these hard peripherals are enabled and configured, corresponding PADS/Fabric pins are exposed at the top level by System Builder when it generates the design.

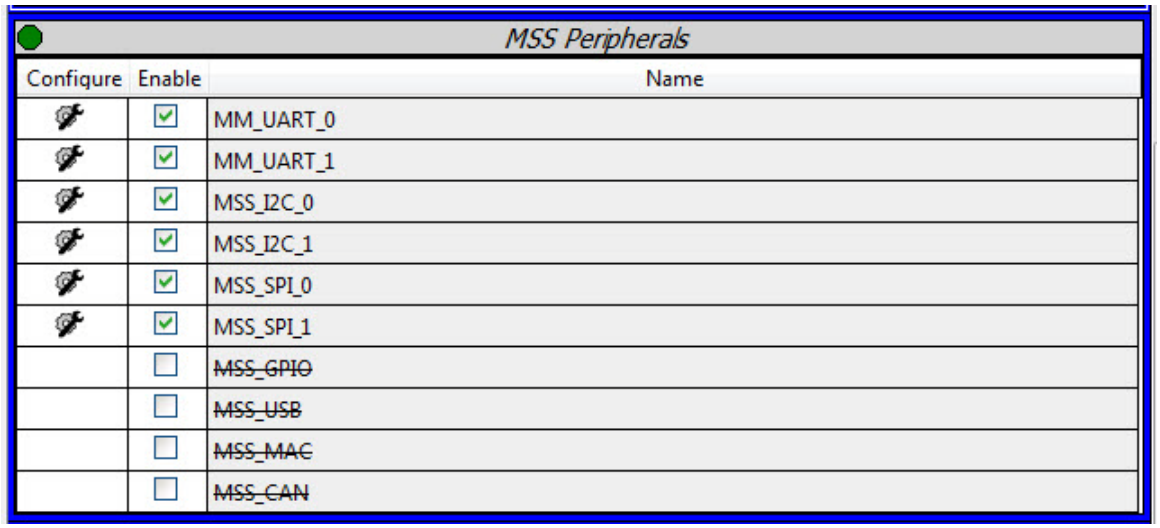

***Figure 2-41* • MSS Peripherals**

System Builder Reset and Miscellaneous Pins

The System Builder block receives and generates reset signals that can be connected to your user logic or board. Depending on your design, you may see the reset pins shown in Table 2-9.

***Table 2-9* • Reset Pins**

Pin	Direction	Functionality and Usage
DEVRST_N (PAD)	In	FPGA Power ON Reset. Deasserts when FPGA is powered up for the first time. This active-low reset performs the same function as a power-up reset. This is an optional function and can safely be tied high in your test bench. When used, this pin must be connected to the dedicated top level port DEVRST_N.
FAB_RESET_N	In	Fabric to System Builder block reset. Can be used by your Fabric logic to reset the system (MSS/SERDES/DDR). This active-low reset performs a re-initialization of all of the peripherals controlled by the MSS. This is an optional function and can safely be tied high.
INIT_PINS:INIT_DONE	Out	Asserts when System Builder block components are ready for communication and initialization is complete. Connect to any fabric logic that needs to wait for System Builder initialization
POWER_ON_RESET_N	Out	When '0', indicates the system controller is in the process of booting. Goes to '0' just after the VDD ramps past the threshold voltage. When '1', indicates the system controller is finished booting.
MSS_READY	Out	Asserts when the MSS is ready for communication with the fabric logic. Connect to User logic's Reset input. MSS_READY happens before the INIT_DONE pins go high. This output only indicates the MSS FICs can be accessed, not that the entire system is operational. To wait until the entire system is operational, wait for MSS _READY and all of the applicable INIT_DONE pins to go high
DDR_READY	Out	This pin is seen only when one/both of MDDR/FDDR are used in the design. Value "1" indicates that both MDDR and FDDR (if used) blocks are ready for communication and their initialization is complete. Connect to any fabric logic that needs to wait for MDDR/FDDR initialization.
SDIF_READY	Out	This pin is seen only when one/more SERDES blocks are used in the design. Value "1" indicates that all the SERDES blocks (if used) are ready for communication and their initialization is complete. Connect to any fabric logic that needs to wait for SERDES initialization.

***Table 2-10* • Miscellaneous Pins**

Pin	Direction	Functionality and Usage
FAB_CCC_PINS: FAB_CCC_GL(0/1/2/3)	Out	Connect to the clock pins of user fabric logic.
FAB_CCC_PINS: FAB_CCC_LOCK	Out	Exposed when FAB_CCC_GL(0/1/2/3) is exposed. When high, FAB_CCC_LOCK signal indicates that the clocks FAB_CCC_GL(0-3) are valid.
CLK(0/1/2/3)_PAD	In	Dedicated Input pad Reference Clock to the Fabric CCC.

***Table 2-10* • Miscellaneous Pins (continued)**

Pin	Direction	Functionality and Usage
CCC_0_PINS: CLK(0/1/2/3)	In	FPGA Fabric Input Reference Clock to the Fabric CCC.
XTL	In	Input source to the External Main Crystal Oscillator.
CHIP_OSC_PINS: XTLOSC_O2F	Out	Crystal Oscillator output. Used to drive user fabric logic or the Fabric CCC.
CHIP_OSC_PINS: RCOSC_1MHZ_O2F	Out	1 MHz RC Oscillator output. Used to drive user fabric logic or the Fabric CCC.
CHIP_OSC_PINS: RCOSC_25_50MHZ_O2F	Out	25/50 MHz RC Oscillator output. Used to drive user fabric logic or the Fabric CCC.
XTLOSC_CCC_OUT	Out/BIF	Crystal Oscillator output. Used to drive the input XTLOSC_CCC_IN BIF pin of Fabric CCC.
RCOSC_1MHZ_CCC_OUT	Out/BIF	1 MHz RC Oscillator output. Used to drive the input RCOSC_1MHZ_CCC_IN BIF pin of Fabric CCC.
RCOSC_25_50MHZ_CCC_OUT	Out/BIF	25/50 MHz RC Oscillator output. Used to drive the input RCOSC_25_50MHZ_CCC_IN BIF pin of Fabric CCC.

System Builder DRC Checks

System Builder enforces Design Rule Checks (DRC) with regard to device-specific peripheral I/O and fabric availability. When the I/Os and/or fabric interface are not available for the specific peripheral/ die/package combination, the Peripherals page of System Builder generates an error and a tooltip message to alert you. You cannot proceed to the next page unless you reconfigure the peripheral appropriately to use supported resources.

In Figure 2-42, the tooltip is generated because the M2S005/144 TQ die/package combination does not support MSS SPI_1 I/Os. You must reconfigure MSS_SPI_1 to use fabric connections instead of I/Os to proceed further.

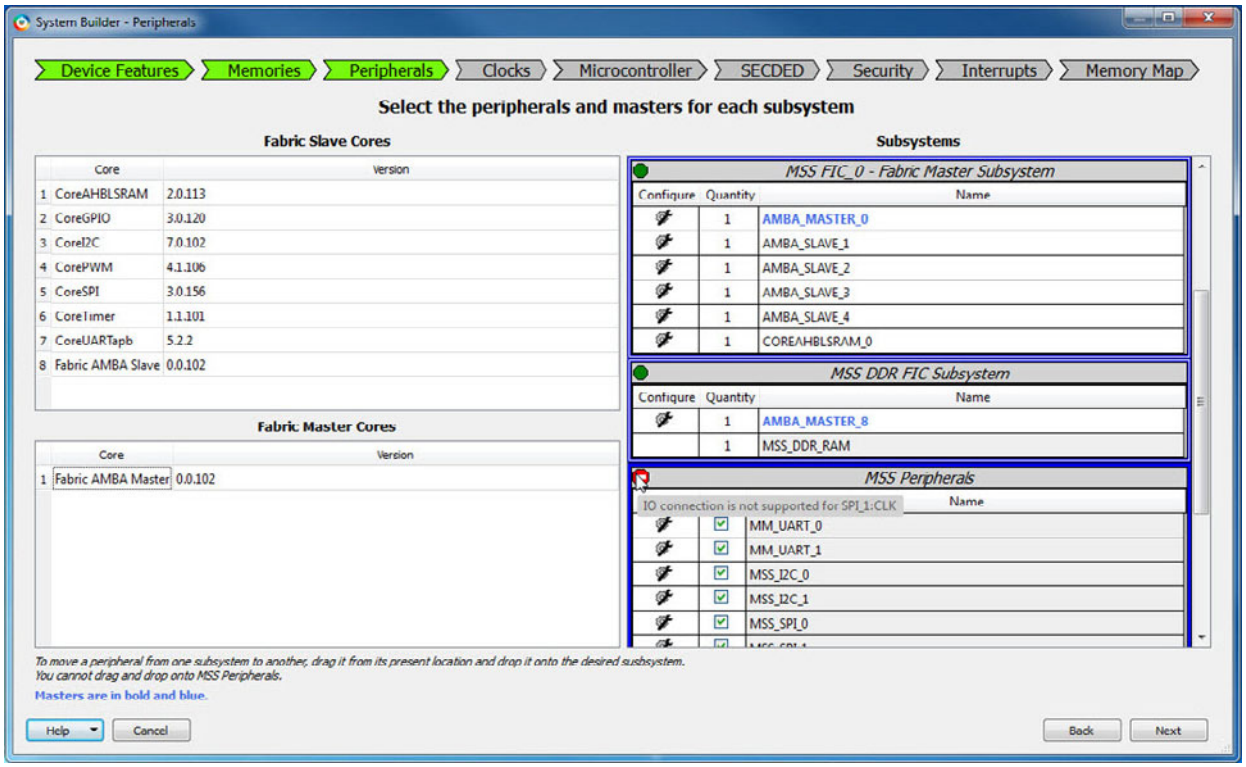

Figure 2-42 • **DRC Check and Tooltip Information**

When you configure multiple MSS peripherals and there are conflicts of shared resources (I/Os or fabrics) between two or more MSS peripherals, the Peripherals page of System Builder displays DRC error messages on the MSS Peripherals group title.

In the example below (Figure 2-43), there is a conflict between MSS Peripheral GPIO and MSS Peripheral I2C. You need to reconfigure either one of the two MSS peripherals to resolve the conflict before you can continue.

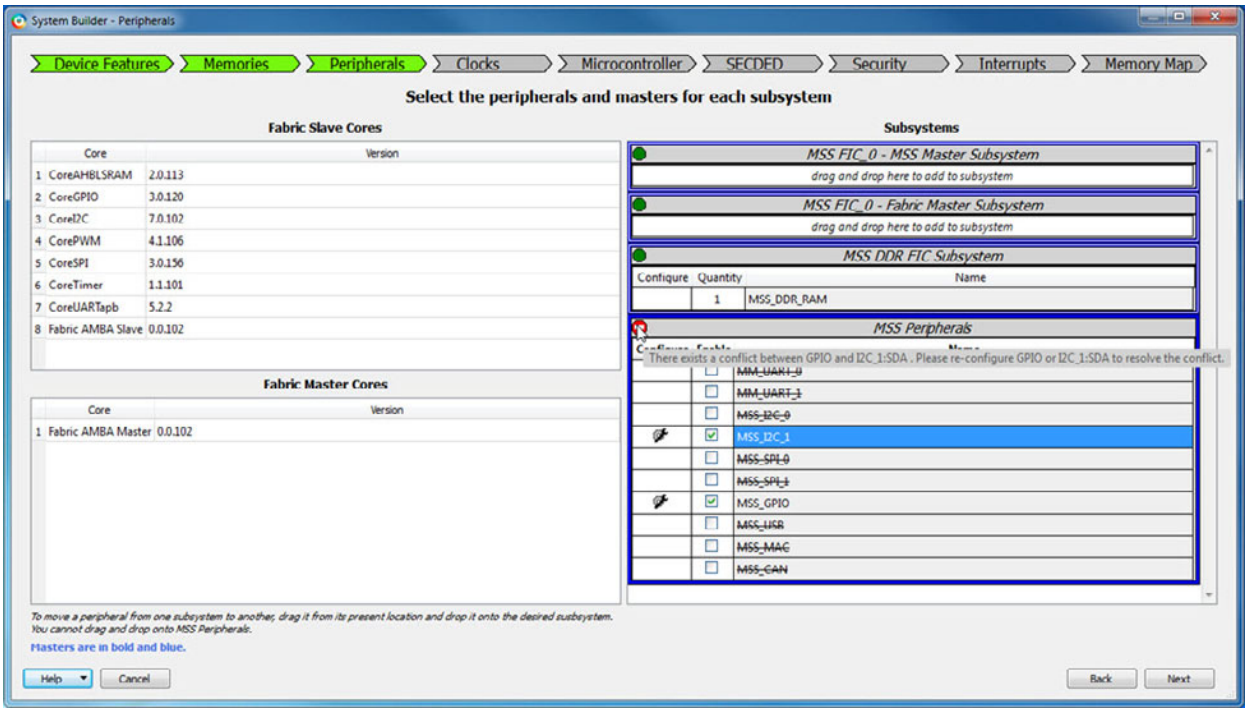

Figure 2-43 • **DRC Error Messages Showing Resource Conflict Between Multiple MSS Peripherals**

If one of the two MSS peripherals in conflict is a GPIO, as in this example, the conflicts (errors and info) are also shown inside the MSS_GPIO configurator (Figure 2-44).

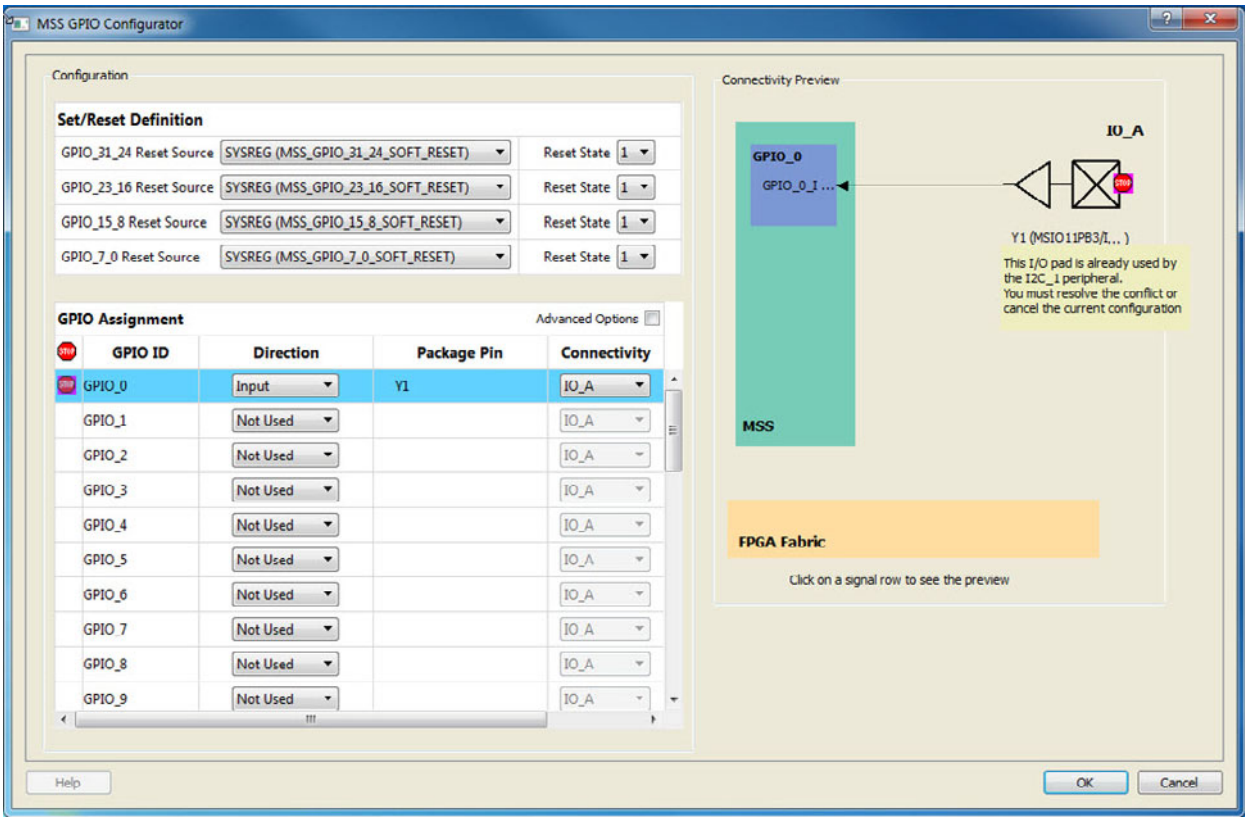

Figure 2-44 • **DRC Conflict Information Displayed in MSS GPIO Configurator**

3 – System Builder Generated Design

After generating your design you will see two new components in your Design Hierarchy. There will be a component named <design_name>, which is a SmartDesign that instantiates your System Builder generated component <design_name>_sb.

The <design_name> is a regular SmartDesign that contains an instance of your System Builder block. Double-click the <design_name>_sb component in your Design Hierarchy or double-click the System Builder instance in the <design_name> SmartDesign canvas to reconfigure the System Builder. When your design is reopened in System Builder, you will observe that all the configuration options you specified while creating the design are preserved.

Figure 3-1 shows the Design Hierarchy after clicking Finish in the final System Builder page on a System Builder named mytop.

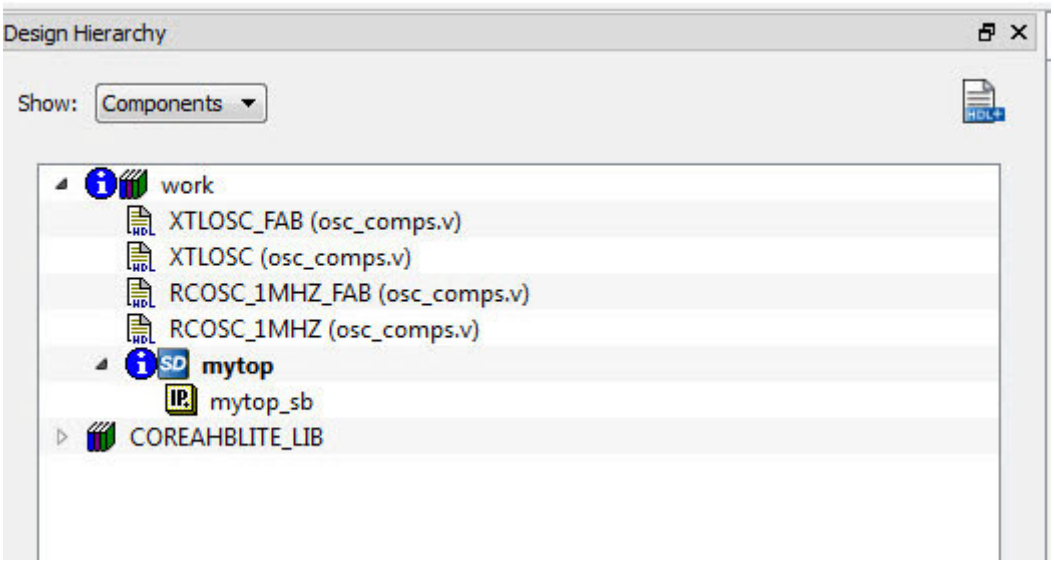

***Figure 3-1* • System Builder Block**

The <design_name> SmartDesign Canvas displays the instance of the System Builder generated component. You can choose to add your own custom peripherals or logic to the <design_name> SmartDesign at this point and connect them to the System Builder instance accordingly.

Your System Builder design is summarized in the System Builder summary report in the Libero Reports View. The name of this file is <design_name>_sb_SYS_BLD.xml. Open the Reports View and click this file for details on what was generated in your System Builder component.

Depending on the type of subsystems you have configured for the System Builder block, the System Builder block may have the ports exposed at the top level, as shown in Figure 3-2 below.

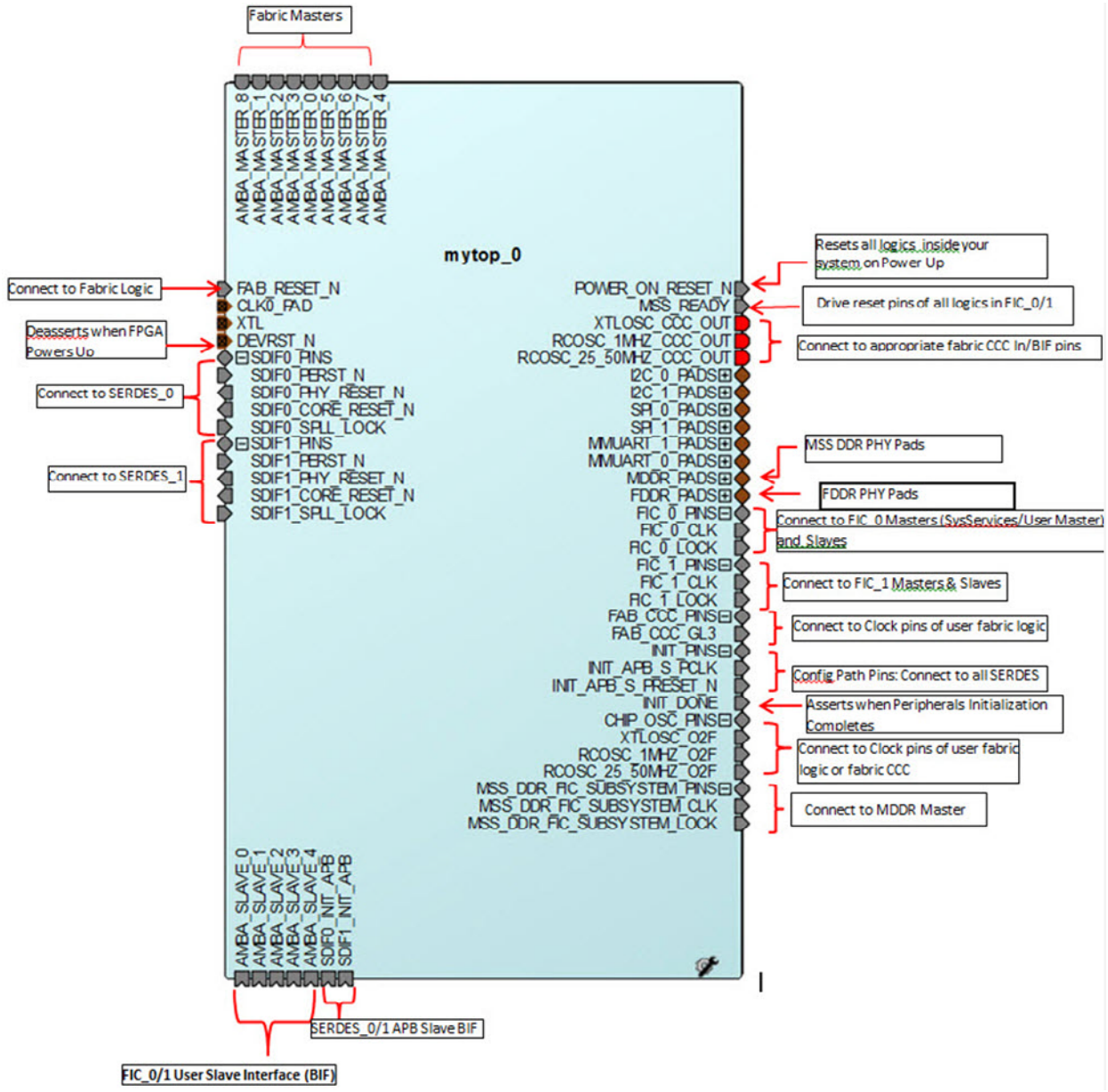

***Figure 3-2* • System Builder Block Top Level Ports (Default Mode - Bus Core Generated Generated inside the System Builder Block)**

4 – Modifying/Inspecting Your System Builder Design

By default, System Builder completes all of the following tasks to build your design:

1. Generates the Initialization circuitry for your peripherals.
2. Generates the Reset circuitry for your peripherals.
3. Generates the Interrupts circuitry (to MSS) from your slaves/peripherals.
4. Instantiates the Clock networks including the CCC per your system requirement.
5. Instantiates and configures the AMBA bus cores for your subsystem per your system requirement.

Note: Items #1 and #2 are excluded when you choose to run System Builder using the Standalone Initialization Methodology (**Project > Project Settings > Design Flow > Use Standalone Initialization for MDDR/FDDR.SERDES peripherals**). For details, refer to the SmartFusion2 Standalone Peripheral Initialization User Guide.

Item #5 is excluded when you choose the Direct Connection Mode (**System Builder > Peripherals > Direct Connection Mode**)

Opening System Builder Block as SmartDesign

To view what is generated inside your System Builder Block or make manual changes at the component level inside the System Builder generated design, you need to open the System Builder Block as SmartDesign. To do so, right-click the System Builder component in the Design Hierarchy and choose **Open as SmartDesign** (Figure 4-1).

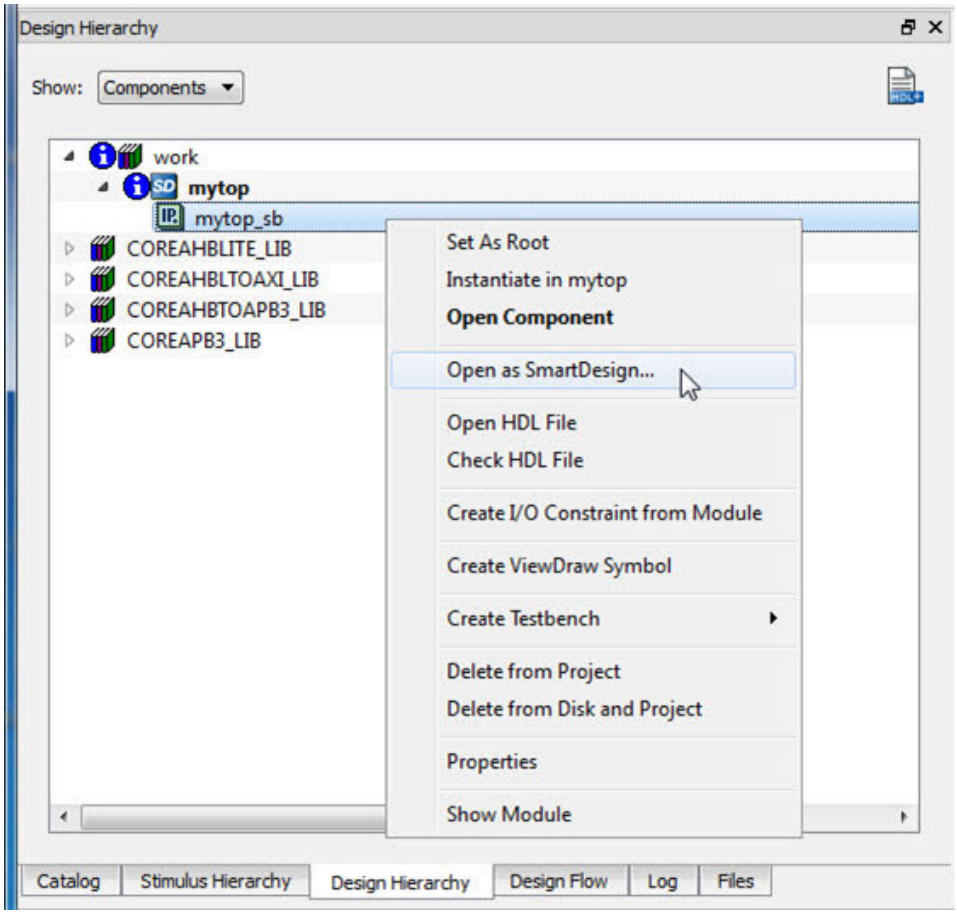

***Figure 4-1* • Open System Builder Component as SmartDesign**

System Builder Block Components

When you open a System Builder block as a SmartDesign, you can see the components System Builder uses to build your design. These include:

- MSS block
- Oscillator Block (FABOSC)
- Clock Conditioning Circuit (CCC)
- Power On Reset macro (SYSRESET_POR)
- CoreConfigP - for Initialization of Peripherals
- CoreResetP - for Initialization of Peripherals
- Peripheral subsystems such as FDDR/MDDR
- AMBA buses for the subsystem per your system requirements.

Manual Changes to System Builder Block

It is not generally necessary to make manual changes to the System Builder block at the component level. However, there are situations where you may want to open the System Builder block as a SmartDesign to make manual changes at the component level. These include but are not limited to the following:

- System Builder allows only one Fabric Master to access the external DDR Memory through the MDDR or FDDR. The System Builder Block it generates has only one AHBLite Master for the MDDR/FDDR. If you want two AHBLite masters to access an external DDR memory via MDDR/FDDR, you can do so by opening the System Builder block as Smart Design, and then open the MDDR configurator or the FDDR configurator to configure two AHBLite master interfaces.
- Depending on the subsystem, System Builder assigns the Peripherals added to that subsystem to the appropriate AMBA bus' slave slots as S0, S1, S2 and so on. The memory mapping is fixed. If you are not happy with the memory mapping and if you want to change it, you can open the System Builder block as SmartDesign, open the AMBA bus configurator and change the slots.
- System Builder does not consider global clock network sharing when driving fabric logic with GLx outputs from the PLL output of the CCC. For each peripheral, a dedicated GLx output from the PLL is used to drive it. System Builder does not implement GLx sharing even though the peripheral subsystems have the same frequency. If some or all of your peripheral subsystems run at the same frequency and therefore may share a PLL GLx output and you want to free up some global GLx outputs from the PLL for other fabric logic, you may want to open the System Builder block as a SmartDesign to do some manual connections and re-wiring. Additionally, from the CCC Configurator's Advanced tab, you have the option to change the phase of the PLL GLx output.

Re-opening the Design in System Builder

If you want to re-open the design in System Builder, right-click the component in the Design Hierarchy and choose **Re-open as System Builder** (Figure 4-2).

WARNING: If you opened the System Builder block as SmartDesign and made manual changes to the design, re-opening the design as System Builder will ignore/overwrite your manual changes. If you want to preserve the manual changes to the design after opening it as SmartDesign, do not re-open it as System Builder.

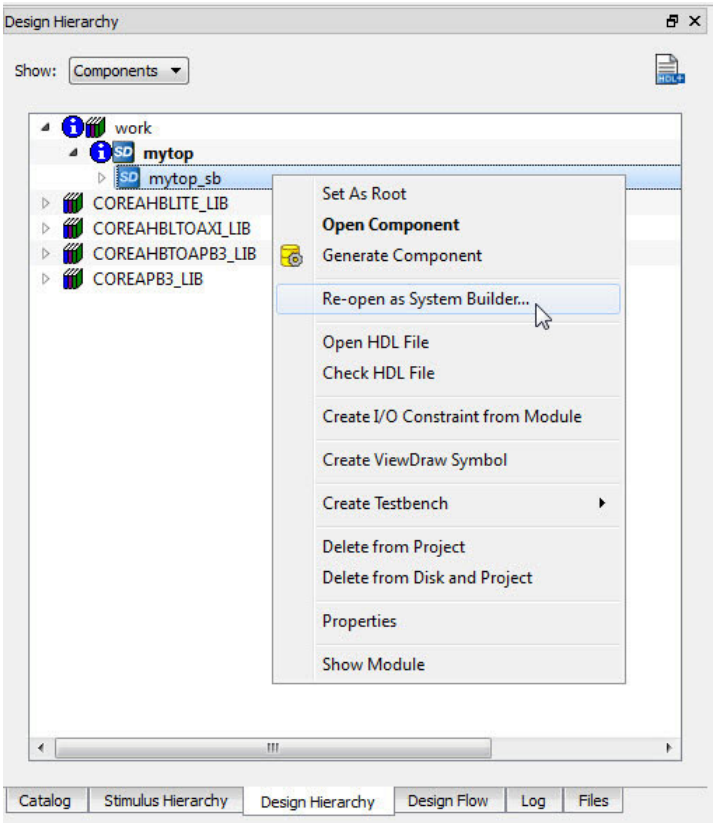

***Figure 4-2* • Re-open as System Builder**

Finishing Your Design

After generating and extending your System Builder design, the rest of the Libero SoC tool flow is the same as for any other normal Libero SoC design.

A – Use Cases

System Builder has the flexibility to build different subsystems per your system and peripheral requirements. For each subsystem, the master may be the Cortex-M3 (MSS) or a fabric master of an AMBA bus type you specify. The fabric slaves are configurable for different AMBA bus types of your choice. Different use cases/scenarios are listed to explain when, how and what subsystem to build for your design needs. For each subsystem, the memory mapping, base addresses and the AMBA bus configuration information is given for you to write your HDL source files or user BFM files for simulation.

Use Case #1

FIC0/1_Fabric_Master_Subsystems with one or more AHBLite fabric masters and different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this subsystem if and when:

- You have one or more (up to a maximum of four) AHBLite fabric masters (either your own user HDL master, a SERDES master or any other fabric master with AHBLite master BIF) which need to access the MSS space (MSS peripherals/eSRAM/eNVM/PDMA/HPDMA) AND
- You want your AHBLite fabric master to master different fabric slaves.

Note: FIC_0 is available in all devices, whereas FIC_1 is available only in the larger devices.

How to Build

In the System Builder Peripherals page:

1. Drag and drop (up to four) Fabric AMBA Master core(s) to the MSS FIC_0/1 - Fabric Master Subsystem.
2. Configure them to be of type AHBLite. The MSS component with its AHBLite slave BIF (FIC_0/1_AHB_SLAVE) will be an inherent slave to these fabric masters.
3. Drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the MSS FIC_0/1 - Fabric Master Subsystem.
4. Configure the count and type (any of AHBLite/AXI/APB3). This will enable the fabric masters you have added earlier to master these fabric slaves.

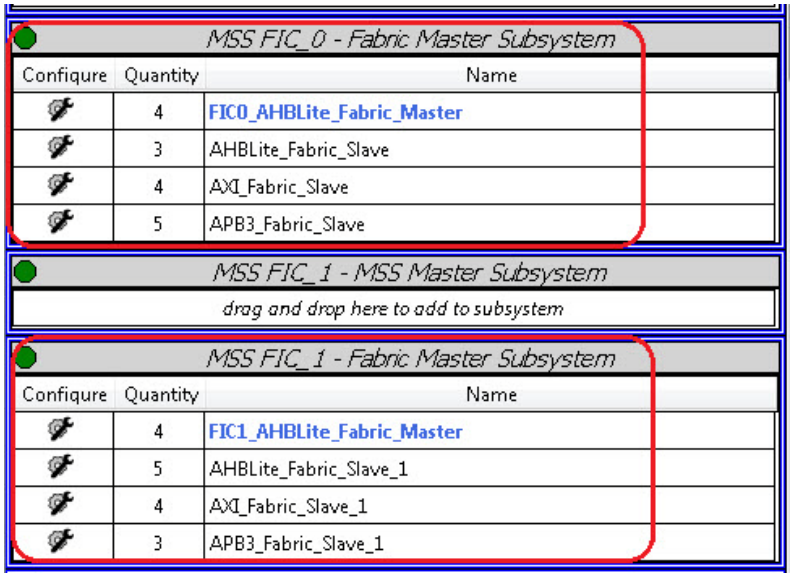

Figure A-1 **• MSS FIC_0/1 - Fabric Master (AHBLite) Subsystems**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to write the user BFM files for simulation.

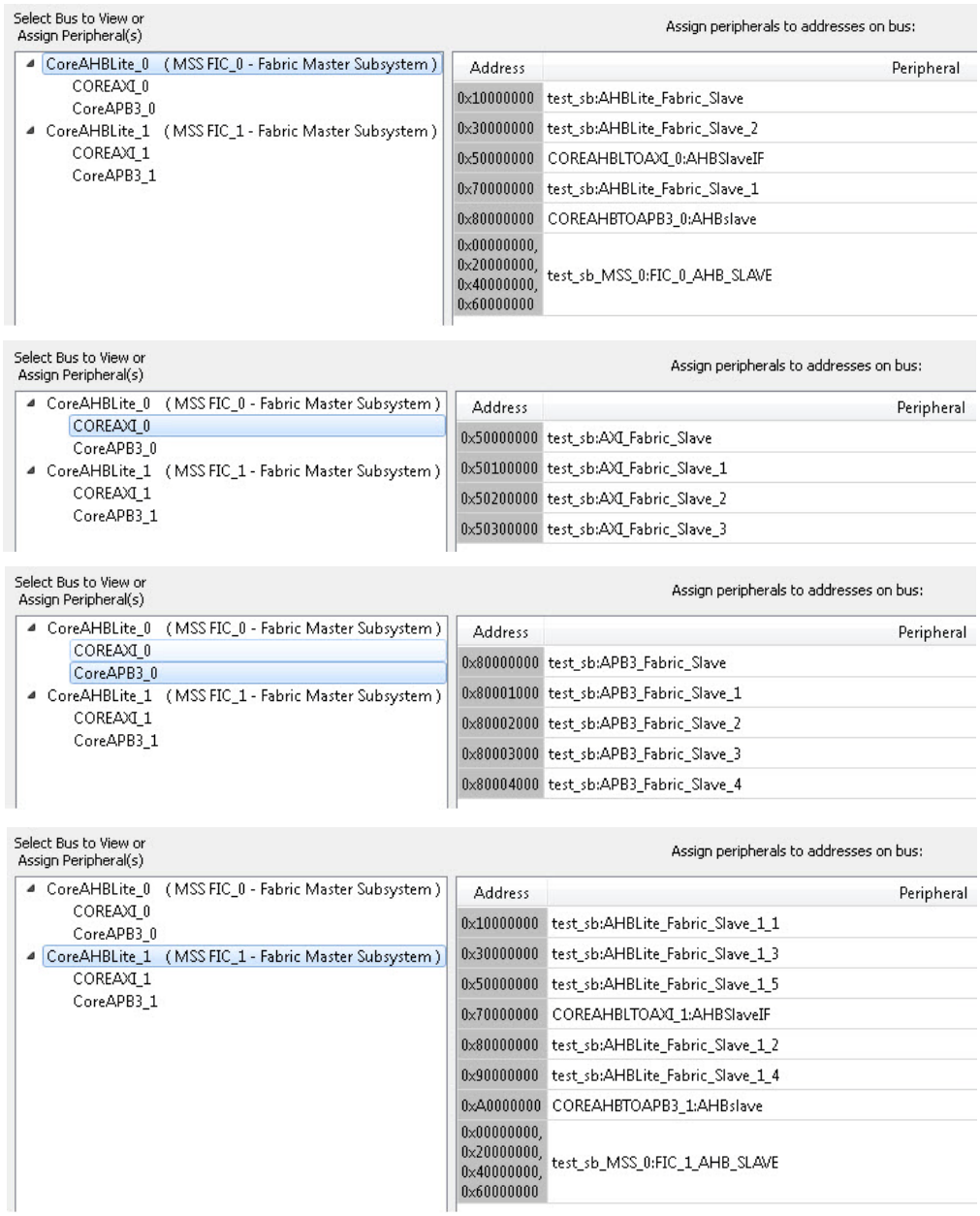

Figure A-2 • **System Builder Memory Map Page**

Understanding the Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in hierarchical order):

- CoreAHBLite - Total 4GB space apportioned into 16 slave slots of 256MB each
  - MSS (FIC_0/1_AHB_SLAVE) is connected as an AHBLite slave to the combined region slave slot of CoreAHBLite
  - Other AHBLite fabric slaves are connected to the normal slave slots of CoreAHBLite
- CoreAXI - Total 16MB space apportioned into 16 slave slots of 1MB each
  - AXI fabric slaves are connected to the slave slots of CoreAXI

- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Top Level Connections in SmartDesign

For all the generic Fabric AMBA Masters and Slaves you add to the MSS FIC_0/1 - Fabric Master Subsystem, System Builder exposes the corresponding Master and Slave BIF ports of appropriate AMBA types with specified names on the System Builder interface. Connect your user fabric Masters and Slaves to appropriate BIF ports in the top level Smart Design component. You can also use the FIC_0/1_CLK and the MSS_READY signals to drive your fabric logic's clock and reset inputs.

Use Case #2

FIC0/1_Fabric_Master_Subsystems with one or more AXI fabric masters and different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this subsystem if and when:

- You have one or more AXI fabric masters (either your own user HDL masters, a SERDES master or any other masters with AXI master BIF) which need to access the MSS space (MSS peripherals/eSRAM/eNVM/PDMA).

  AND

- You also want your AXI fabric masters to master different fabric slaves.

Note: FIC_0 is available in all devices whereas FIC_1 is available only in the bigger devices.

How to Build

In the System Builder Peripherals page:

1. Drag and drop (up to four) Fabric AMBA Master core(s) to the MSS FIC_0/1 - Fabric Master Subsystem.
2. Configure them to be of type AXI. The MSS component with its AHBLite slave BIF (FIC_0/1_AHB_SLAVE) will be an inherent slave to these fabric masters (connected via CoreAXI to CoreAXItoAHBL to CoreAHBLite).
3. Drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the MSS FIC_0/1 - Fabric Master Subsystem,
4. Configure their count and type (AHBLite/AXI/APB3). This will enable the fabric masters you have added earlier to master these fabric slaves.

***Figure A-3* • MSS FIC_0/1 - Fabric Master (AXI) Subsystems**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

.

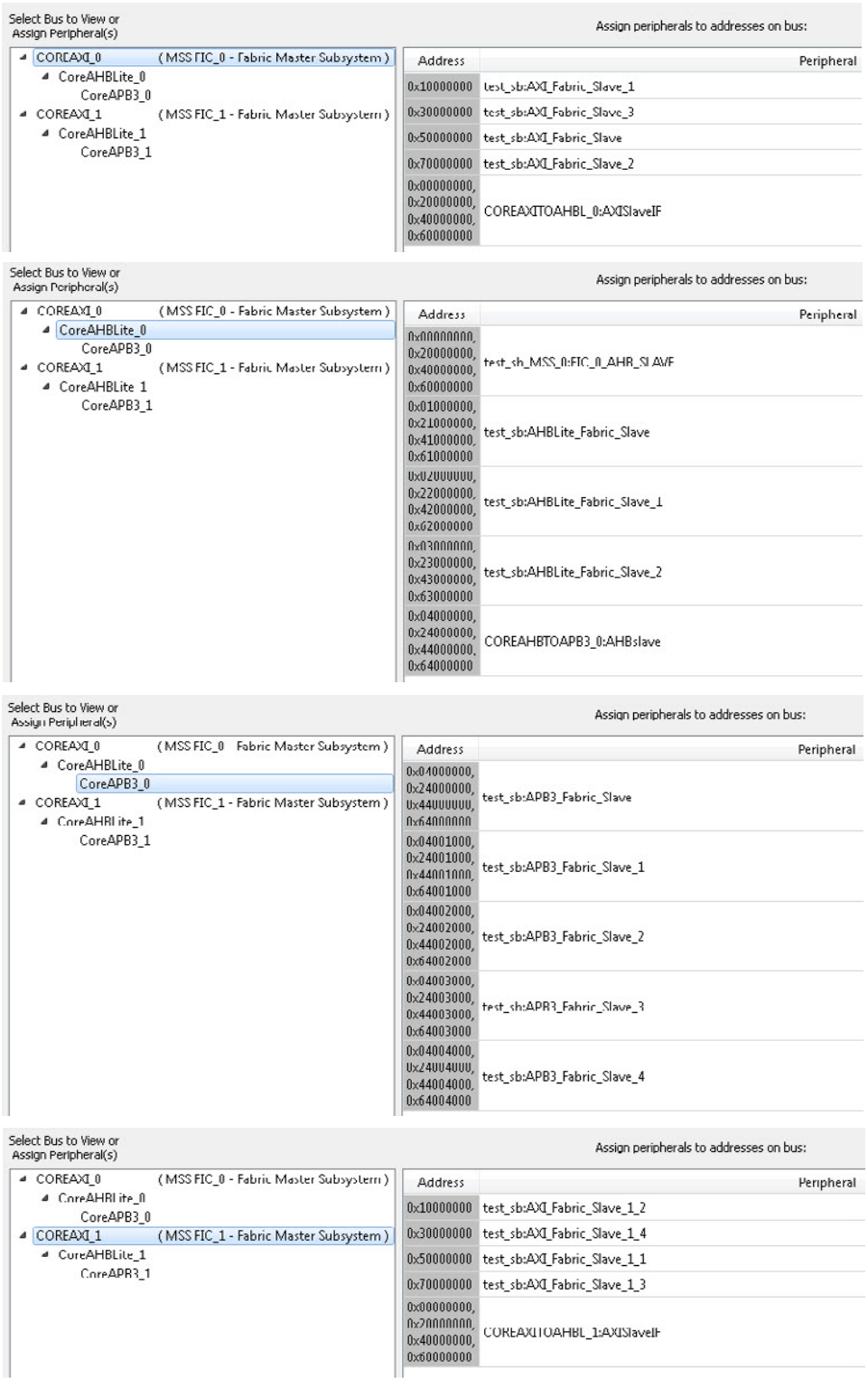

Figure A-4 • **System Builder Memory Map Page**

Understanding Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in hierarchical order):

- CoreAXI - Total 4GB space apportioned into 16 slave slots of 256MB each
  - CoreAHBLite is connected to the combined region slave slot of CoreAXI via CoreAXItoAHBL
  - Other AXI fabric slaves are connected to the normal slave slots of CoreAXI
- CoreAHBLite - Total 256MB space apportioned into 16 slave slots of 16MB each
  - MSS (FIC_0/1_AHB_SLAVE) is connected to the slave slot 0 of CoreAHBLite
  - Other AHBLite fabric slaves are connected to the other slave slots of CoreAHBLite
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Connections in the Top Level SmartDesign

For all the generic Fabric AMBA Masters and Slaves added to the MSS FIC_0/1 - Fabric Master Subsystem, System Builder exposes the corresponding Master and Slave BIF ports of appropriate types with specified names on the System Builder interface. Connect your user fabric Masters and Slaves to appropriate BIF ports in the top level Smart Design component. You can also use the FIC_0/1_CLK and the MSS _READY signals to drive your fabric logic's clock and reset inputs.

Use Case #3

FIC0/1_MSS _Master_Subsystems with different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this subsystem if and when you want the Cortex M3 (MSS) to master different fabric slaves (via FIC_0/1), all operating at the same frequency (FIC_0/1_CLK) and controlled by the same reset signal - MSS_READY.

Note: FIC_0 is available in all devices whereas FIC_1 is available only in the bigger devices.

How to Build

In the System Builder Peripherals page:

1. Drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the MSS FIC_0/1 - MSS Master Subsystem.
2. Configure their count and type (AHBLite/AXI/APB3). The Cortex-M3 is the inherent master to all these fabric slaves you add.

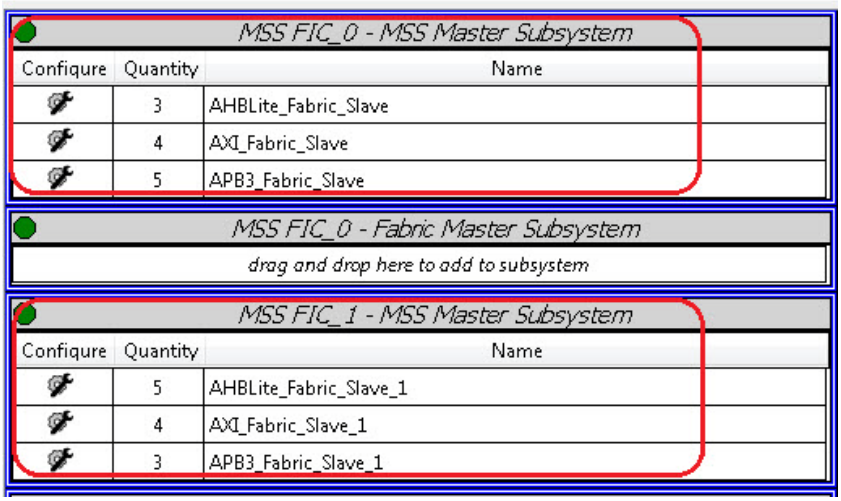

***Figure A-5* • MSS FIC_0/1 - MSS Master Subsystem**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to write the user BFM files for simulation.

.

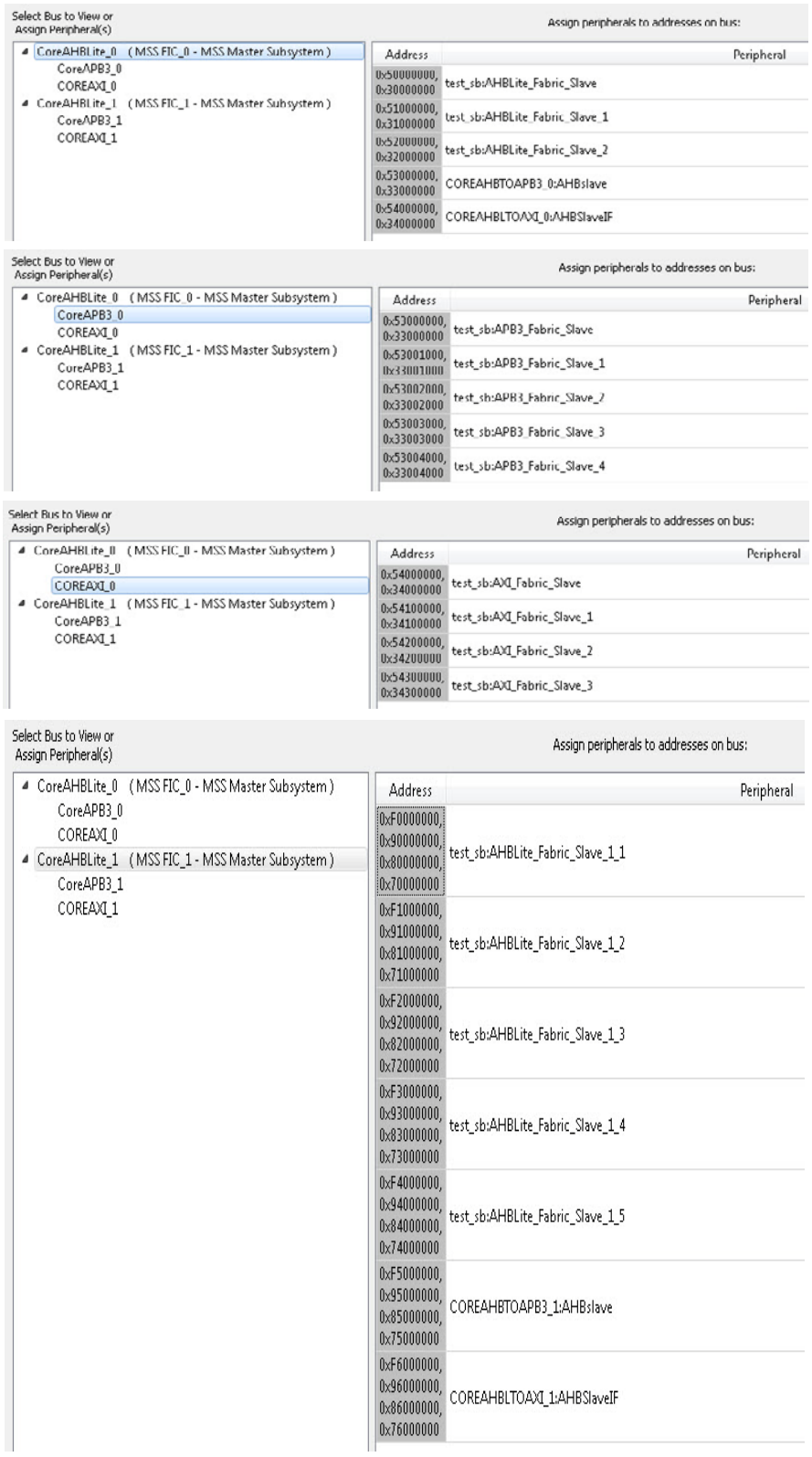

***Figure A-6* • System Builder Memory Map Page**

Understanding Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in hierarchical order):

- CoreAHBLite - Total 256MB space apportioned into 16 slave slots of 16MB each
  - The FIC_0/1_AHB_MASTER BIF of MSS masters the CoreAHBLite and drives the upper 4 bits of the address based on whether it is FIC0 or FIC1
  - AHBLite fabric slaves are connected to the slave slots of CoreAHBLite
- CoreAXI - Total 16MB space apportioned into 16 slave slots of 1MB each
  - AXI fabric slaves are connected to the slave slots of CoreAXI
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Top Level Connections in SmartDesign

For all the generic Fabric AMBA Slaves added to the MSS FIC_0/1 - MSS Master Subsystem, System Builder exposes the corresponding slave BIF ports of appropriate types with specified names on the System Builder interface. Connect your user fabric slaves to the appropriate BIF ports in the top level Smart Design component. You can also use the FIC_0/1_CLK and the MSS_READY signals to drive your fabric logic's clock and reset inputs.

Use Case #4

MSS DDR_FIC_Subsystem with one AHBLite fabric master and different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this system when:

- You have one AHBLite fabric master (either your own user HDL master, a SERDES master or any other master with AHBLite master BIF) which needs to access external DDR memory via the MDDR

  AND

- You want your AHBLite fabric master to master different other fabric slaves in addition to the MSS_DDR_RAM.

How to Build

In the System Builder Peripherals page, drag and drop 1 Fabric AMBA Master core to the MSS DDR FIC Subsystem. Configure it to be of type AHBLite. The MDDR with its AHBLite slave BIF (MDDR_DDR_AHB0_SLAVE) will be an inherent slave to the fabric master. In addition, drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the MSS DDR FIC Subsystem. Configure their count and type (any of AHBLite/AXI/APB3). This would enable the fabric master you have added earlier to master these fabric slaves.

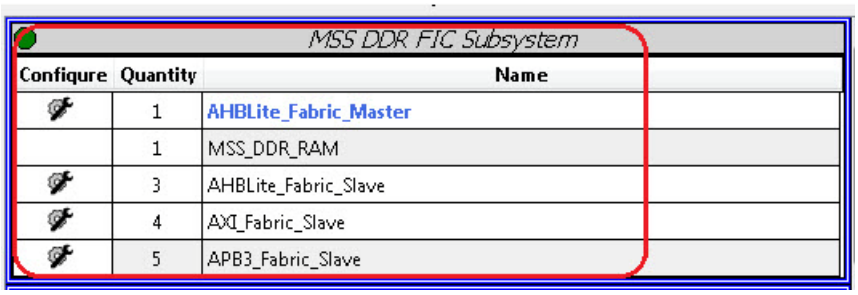

Figure A-7 • **MSS DDR_FIC Subsystem**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

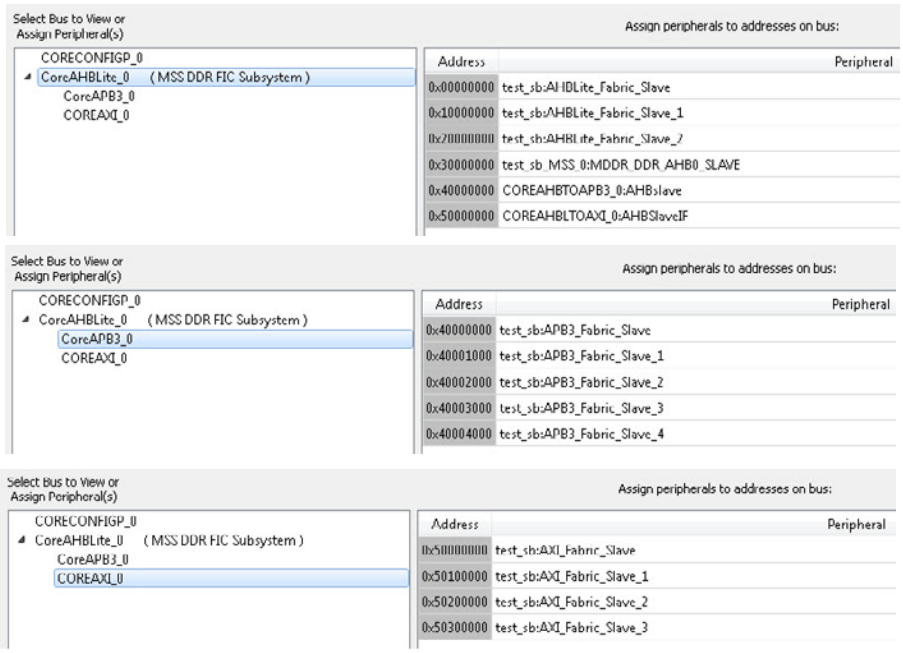

***Figure A-8* • System Builder Memory Map Page**

Understanding Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAHBLite - Total 4GB space apportioned into 16 slave slots of 256MB each
  - MDDR_DDR_AHB0_SLAVE BIF of the MSS is connected as an AHBLite slave to one of the slave slots of CoreAHBLite
  - Other AHBLite fabric slaves are connected to the other slave slots of CoreAHBLite
- CoreAXI - Total 16MB space apportioned into 16 slave slots of 1MB each
  - AXI fabric slaves are connected to the slave slots of CoreAXI
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Top Level Connections in SmartDesign

For all the generic Fabric AMBA Master and Slaves added to the MSS DDR_FIC Subsystem, System Builder exposes corresponding master and slave BIF ports of appropriate types with specified names on the System Builder interface. Connect your user fabric master and slaves to appropriate BIF ports in the top level Smart Design component. You can also use the DDR/SMC_FIC_CLK and the MSS_READY, DDR_READY or INIT_DONE signals to drive your fabric logic's clock and reset inputs, respectively.

Use Case #5

MSS DDR_FIC_Subsystem with one AXI fabric master and different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this subsystem if and when:

- You have one AXI fabric master (either your own user HDL master, a SERDES master or any other master with AXI master BIF) which needs to access external DDR memory via MDDR

  AND

- You want your AXI fabric master to master different other fabric slaves in addition to the MSS_DDR_RAM.

How to Build

In the System Builder Peripherals page, drag and drop one Fabric AMBA Master core to the MSS DDR FIC Subsystem. Configure it to be of type AXI. The MDDR with its AXI slave BIF (MDDR_DDR_AXI_SLAVE) will be an inherent slave to the fabric master. In addition, drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the MSS DDR FIC Subsystem. Configure their count and type (any of AHBLite/AXI/APB3). This would enable the fabric master you have added earlier to master these fabric slaves.

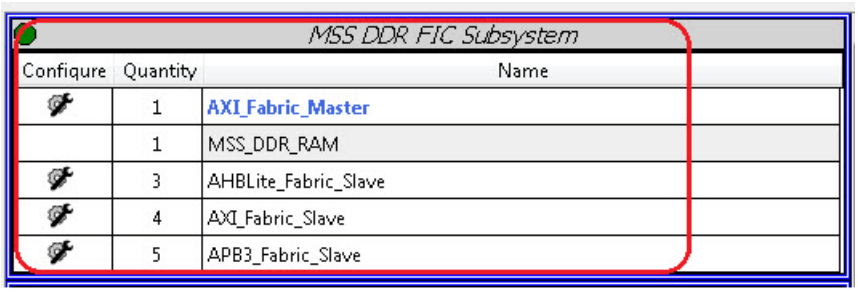

Figure A-9 • **MSS DDR_FIC Subsystem**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

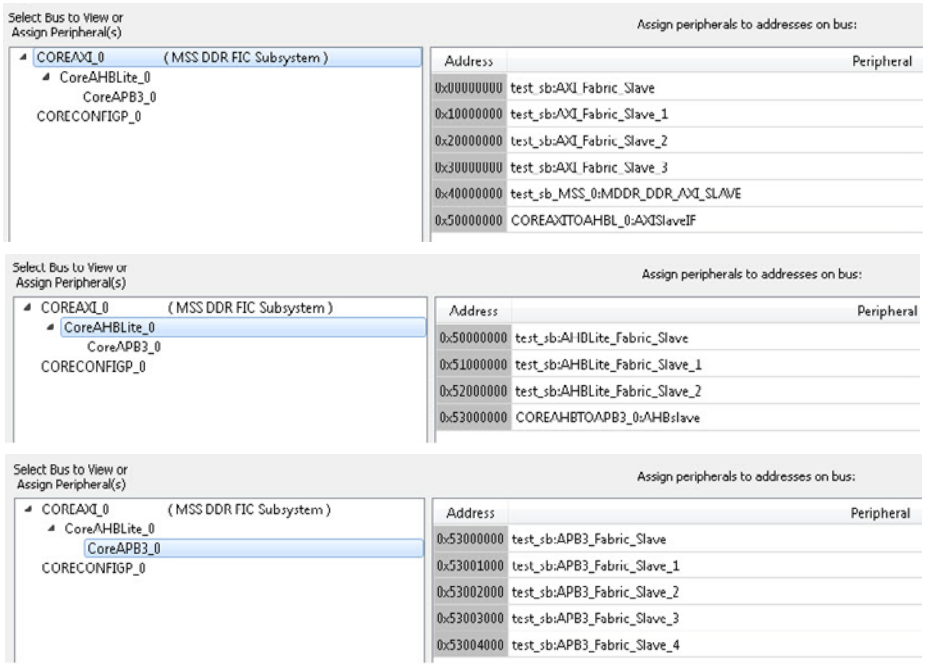

***Figure A-10* • System Builder Memory Map Page**

Understanding Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAXI - Total 4GB space apportioned into 16 slave slots of 256MB each
  - MDDR_DDR_AXI_SLAVE BIF of the MSS is connected as an AXI slave to one of the slave slots of CoreAXI
  - Other AXI fabric slaves are connected to the other slave slots of CoreAXI
- CoreAHBLite - Total 256MB space apportioned into 16 slave slots of 16MB each
  - AHBLite fabric slaves are connected to the slave slots of CoreAHBLite
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Top Level Connections in SmartDesign

For all the generic Fabric AMBA Master and Slaves added to the MSS DDR FIC Subsystem, System Builder exposes corresponding master and slave BIF ports of appropriate types with specified names on the System Builder interface. Connect your user fabric master and slaves to appropriate BIF ports in the top level Smart Design component. You can also use the DDR/SMC_FIC_CLK and the MSS_READY, DDR_READY or INIT_DONE signals to drive your fabric logic's clock and reset inputs, respectively.

Use Case #6

MSS DDR_FIC_Subsystem with one AXI fabric master and no other slaves

When to Build

You build this subsystem if and when

- You have one AXI fabric master (either your own user HDL master, a SERDES master or any other master with AXI master BIF) which needs to access the external DDR memory via MDDR, AND
- Your AXI fabric master needs to access the external DDR memory without any restrictions on the addressable space normally imposed by the bus core's slot size configurations. This is achieved by taking advantage of the CoreAXI's 'FEED THROUGH' mode, AND
- You don't need any fabric slaves in the subsystem other than the DDR memory.

How to Build

In the System Builder Peripherals page, drag and drop one Fabric AMBA Master core to the MSS DDR FIC Subsystem. Configure it to be of type AXI. The MDDR with its AXI slave BIF (MDDR_DDR_AXI_SLAVE) will be an inherent slave to the fabric master. Do not add any other master/slave cores to this subsystem.

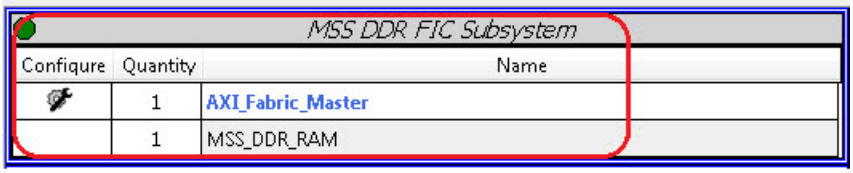

Figure A-11 • **MSS DDR_FIC Subsystem**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

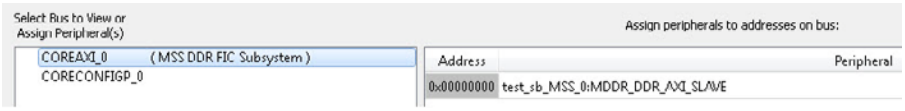

Figure A-12 • **System Builder Memory Map**

Understanding the Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAXI - Configured in 'FEED THROUGH' mode.
  - FEED THROUGH' is like a direct wire connection where the CoreAXI's master BIF M0 is directly connected to its slave BIF S0, without any restrictions of address space or slot sizes. This enables the AXI fabric master to access the external DDR memory space without any restrictions.
  - MDDR_DDR_AXI_SLAVE BIF of the MSS is connected as an AXI slave to the slave slot 0 of CoreAXI

Making the Top Level Connections in SmartDesign

For the generic Fabric AMBA Master added to the MSS DDR FIC Subsystem, System Builder exposes corresponding master BIF port on the System Builder interface. Connect your user fabric master to this BIF port in the top level Smart Design component. You can also use the DDR/SMC_FIC_CLK and the MSS_READY, DDR_READY or INIT_DONE signals to drive your fabric logic's clock and reset inputs, respectively.

Use Case #7

Fabric_DDR_Subsystem with one AHBLite fabric master and different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this system if and when:

- You have one AHBLite fabric master (either your own user HDL master, SERDES master or any other master with AHBLite master BIF) which needs to access external DDR memory via FDDR

  AND

- You want your AHBLite fabric master to master different other fabric slaves in addition to the Fabric_DDR_RAM.

How to Build

In the System Builder Peripherals page, drag and drop one Fabric AMBA Master core to the Fabric DDR Subsystem. Configure it to be of type AHBLite. The FDDR with its AHBLite slave BIF (AHB0_SLAVE) will be an inherent slave to the fabric master. In addition, drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the Fabric DDR Subsystem. Configure their count and type (any of AHBLite/AXI/APB3). This would enable the fabric master you have added earlier to master these fabric slaves.

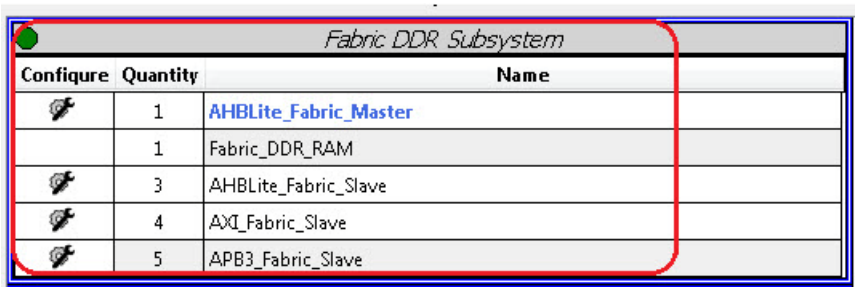

Figure A-13 **• System Builder Fabric DDR Subsystem**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

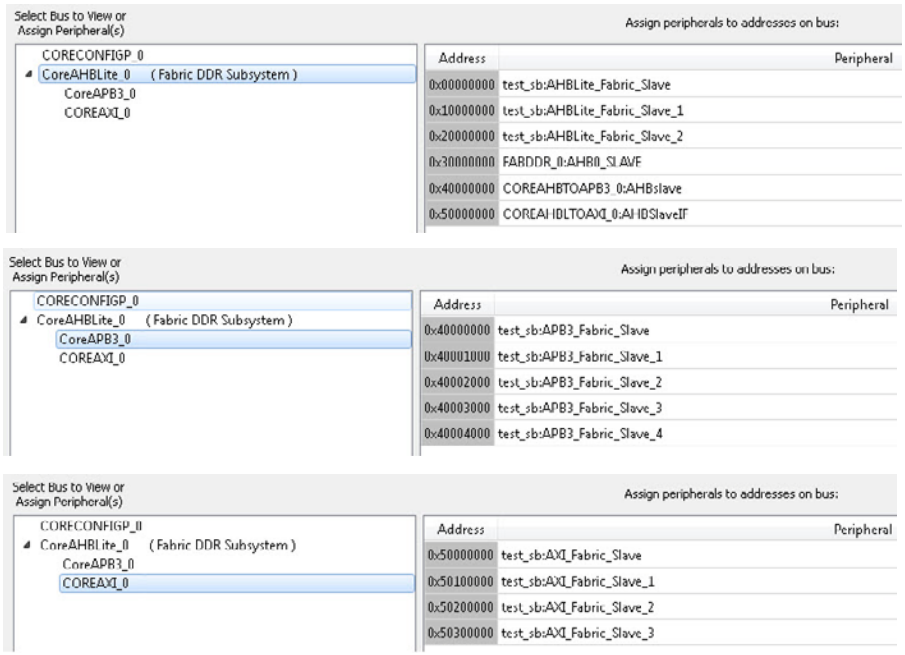

Figure A-14 • **System Builder Memory Map Page**

Understanding Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAHBLite - Total 4GB space apportioned into 16 slave slots of 256MB each
  - AHB0_SLAVE BIF of the FDDR is connected as an AHBLite slave to one of the slave slots of CoreAHBLite
  - Other AHBLite fabric slaves are connected to the other slave slots of CoreAHBLite
- CoreAXI - Total 16MB space apportioned into 16 slave slots of 1MB each
  - AXI fabric slaves are connected to the slave slots of CoreAXI
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Top Level Connections in SmartDesign

For all the generic Fabric AMBA Master and Slaves added to the Fabric DDR Subsystem, System Builder exposes corresponding master and slave BIF ports of appropriate types with specified names on the System Builder interface. Connect your user fabric master and slaves to appropriate BIF ports in the top level Smart Design component. You can also use the FDDR_SUBSYSTEM_CLK and the MSS_READY, DDR_READY or INIT_DONE signals to drive your fabric logic's clock and reset inputs, respectively.

Use Case #8

Fabric_DDR_Subsystem with one AXI fabric master and different types of slaves (AXI/AHBLite/APB3)

When to Build

You build this system if and when:

- You have one AXI fabric master (either your own user HDL master, a SERDES master or any other master with AXI master BIF) which needs to access external DDR memory via FDDR.

  AND

- You want your AXI fabric master to master different other fabric slaves in addition to the Fabric_DDR_RAM.

How to Build

In the System Builder Peripherals page, drag and drop one Fabric AMBA Master core to the Fabric DDR Subsystem. Configure it to be of type AXI. The FDDR with its AXI slave BIF (AXI_SLAVE) will be an inherent slave to the fabric master. Drag and drop any of the Fabric Slave Cores (the ones with hard-coded core versions or the generic Fabric AMBA Slaves) to the Fabric DDR Subsystem, configure their count and type (any of AHBLite/AXI/APB3). This would enable the fabric master you have added earlier to master these fabric slaves.

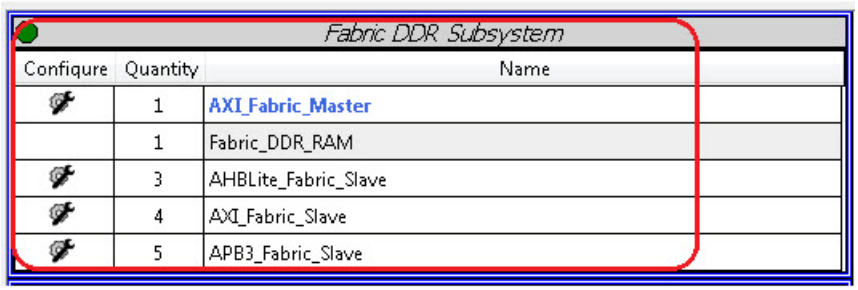

Figure A-15 • **System Builder Fabric DDR Subsystem**

Checking the Memory Map

The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

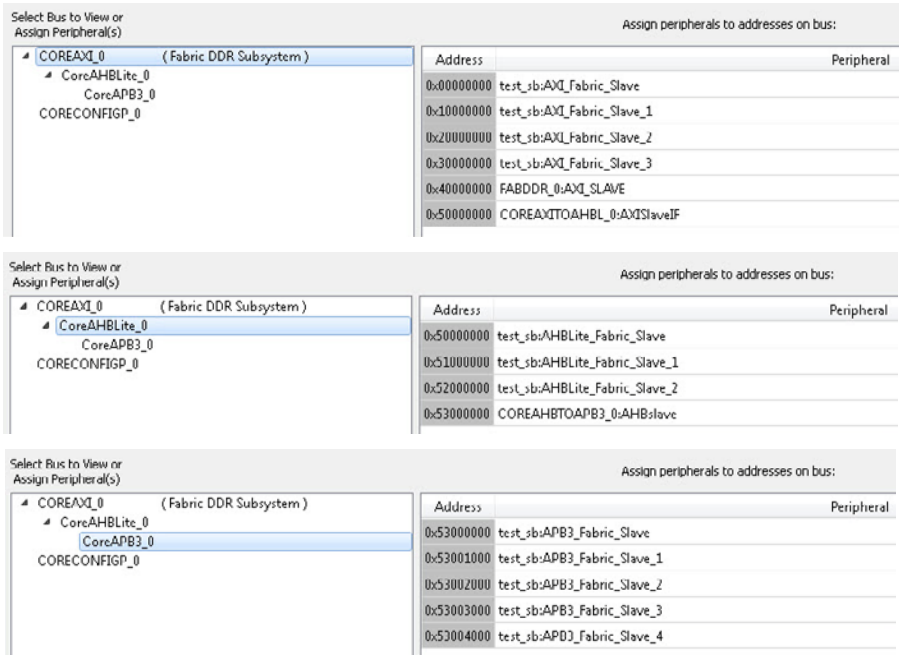

Figure A-16 • **System Builder Memory Map Page**

Understanding Bus Core Configuration

As shown in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAXI - Total 4GB space apportioned into 16 slave slots of 256MB each
  - AXI_SLAVE BIF of the FDDR is connected as an AXI slave to one of the slave slots of CoreAXI
  - Other AXI fabric slaves are connected to the other slave slots of CoreAXI
- CoreAHBLite - Total 256MB space apportioned into 16 slave slots of 16MB each
  - AHBLite fabric slaves are connected to the slave slots of CoreAHBLite
- CoreAPB3 - Total 64KB space apportioned into 16 slave slots of 4KB each
  - APB3 fabric slaves are connected to the slave slots of CoreAPB3

Making the Top Level Connections in SmartDesign:

For all the generic Fabric AMBA Master and Slaves added to the Fabric DDR Subsystem, System Builder exposes corresponding master and slave BIF ports of appropriate types with specified names on the System Builder interface. Connect your user fabric master and slaves to appropriate BIF ports in the top level Smart Design component. You can also use the FDDR_SUBSYSTEM_CLK and the MSS_READY, DDR_READY or INIT_DONE signals to drive your fabric logic's clock and reset inputs, respectively.

Use Case #9

Fabric_DDR_Subsystem with one AXI fabric master and no other slaves

When to Build

You build this system if and when:

- You have one AXI fabric master (either your own user HDL master, a SERDES master or any other master with AXI master BIF) which needs to access the external DDR memory via FDDR (Fabric DDR), AND
- Your AXI fabric master needs to access the external DDR memory without any restrictions on the addressable space normally imposed by the bus core's slot size configurations. This is achieved by taking advantage of the CoreAXI's 'FEED THROUGH' mode, AND
- You don't need any fabric slaves in the subsystem other than the DDR memory.

How to Build

In the System Builder Peripherals page, drag and drop one Fabric AMBA Master core to the Fabric DDR Subsystem. Configure it to be of type AXI. The FDDR with its AXI slave BIF (AXI_SLAVE) will be an inherent slave to the fabric master. Do not add any other master/slave cores to this subsystem.

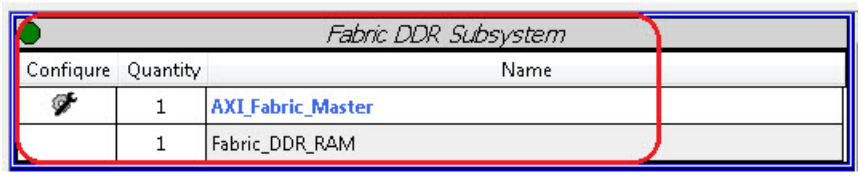

Figure A-17 • **System Builder Fabric DDR Subsystem**

Check the Memory Map

For the fabric master to access the external DDR memory space, you should know the bus core configuration and the memory map of the Fabric_DDR_RAM in the subsystem. The System Builder Memory Map page is where you can view the memory map of different subsystems in use and the base addresses of different slaves. This is useful information for you to address correct memory spaces in your HDL source files and write the user BFM files for simulation.

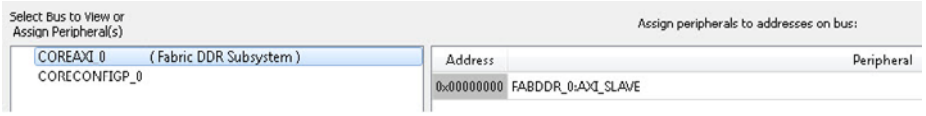

Figure A-18 • **System Builder Memory Map Page**

Bus Core Configuration

As can be observed in the memory map, the following is the bus core configuration (listed in the hierarchical order):

- CoreAXI ' Configured in 'FEED THROUGH' mode.
  - FEED THROUGH' is like direct wire connection where the CoreAXI's master BIF M0 is directly connected to its slave BIF S0, without any restrictions of address space or slot sizes. This enabled the AXI fabric master to access the external DDR memory space without any restrictions.
  - AXI_SLAVE BIF of the FDDR is connected as an AXI slave to the slave slot 0 of CoreAXI

Making the Top Level Connections in SmartDesign

For the generic Fabric AMBA Master added to the Fabric DDR Subsystem, System Builder exposes corresponding master BIF port on the System Builder interface. Connect your user fabric master to this BIF port in the top level Smart Design component. You can also use the FDDR_SUBSYSTEM_CLK and the MSS_READY, DDR_READY or INIT_DONE signals to drive your fabric logic's clock and reset inputs, respectively.

B – Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

Customer Service

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**

Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Technical Support

For Microsemi SoC Products Support, visit http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support.

Website

You can browse a variety of technical and non-technical information on the Microsemi SoC Products Group home page, at www.microsemi.com/soc.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office.

Visit About Us for sales office listings and corporate contacts.

Sales office listings can be found at www.microsemi.com/soc/company/contact/default.aspx.

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

E-mail: sales.support@microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace 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, Calif. and has approximately 4,800 employees globally. Learn more at **www.microsemi.com**.

5-02-00378-5/05.16