SmartFusion2 FPGA

Microcontroller Subsystem BFM Simulation Guide

Microsemi

Table of Contents

Introduction

The SmartFusion2 FPGA's Microcontroller Subsystem (MSS) can be simulated using ModelSim or other supported third-party simulators. MSS Simulation is performed using a Bus Functional Model (BFM) strategy. Simulation can be useful in the following situations:

- Verifying the connectivity and addressing of MSS peripherals.
- Verifying the DDR Memory configuration and addressing with your vendor's memory (subject to availability of appropriate HDL memory models).
- Verifying addressing of peripherals in the Fabric that are connected to the MSS using the 32-bit or 64-bit Fabric Interface Controllers (FICs).

This document describes how to simulate your SmartFusion2 FPGA design that includes the MSS.

Note: Only one MSS component is allowed in your design.

1 – Simulation Models

Cortex-M3 BFM

The SmartFusion2 MSS' Cortex-M3 processor is modeled with Microsemi's AMBA Bus Functional Model (BFM). Refer to Microsemi's DirectCore AMBA BFM User's Guide for details on the supported instructions and syntax of the BFM commands.

MSS Peripherals

To minimize simulation time, certain peripherals in the SmartFusion2 MSS do not have full behavioral models. Instead they are replaced with memory models that will output a message indicating when the memory locations inside the peripheral have been accessed. This means that the peripheral output signals do not toggle based on any writes to registers, or react to any signal inputs on the protocol pins. The peripherals without full behavioral models are:

- CAN
- Ethernet
- MMUART
- I2C
- PDMA
- RTC
- SPI
- USB
- WatchDog

The peripherals that have full behavioral models are:

- AHB Bus Matrix
- eNVM
- Fabric Interface Controllers
- GPIO
- MDDR
- MSS Clock Conditioning Circuit

Fabric Peripherals

RTL level simulation models are available to simulate Fabric peripherals.

2 – Simulation Flow

Figure 2-1 illustrates the hierarchy of a SmartFusion2 design that includes the MSS and two fabric peripherals. The MSS component is instantiated in a top level SmartDesign component with fabric peripherals. The fabric peripherals 1 and 2 are two instances of CoreGPIO (32-bits each). In this scenario, generating the MSS component produces the following four *bfm files for simulation. These BFM files are generated in the <project_dir>/simulation folder

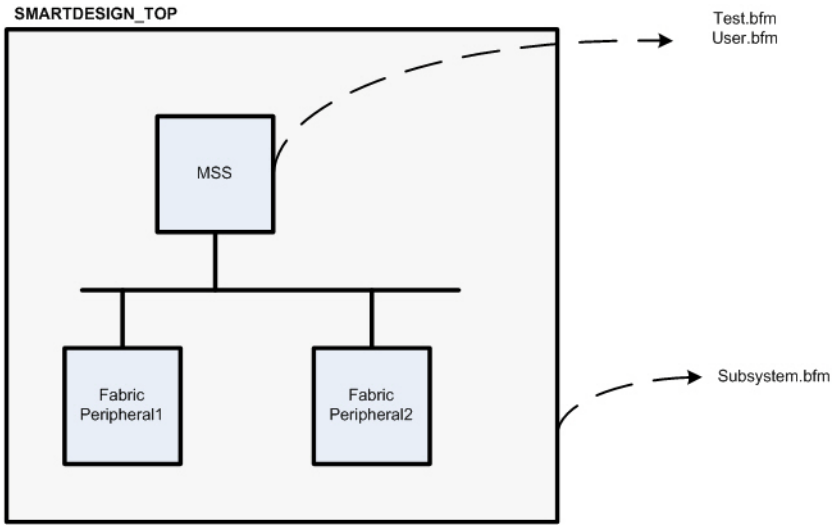

Figure 2-1 **• Example MSS Design**

test.bfm - Contains the BFM commands to initialize the simulation model. The BFM commands in this file are generated based upon your MSS configuration. This file is analogous to the system boot code, as it initializes the MSS and calls your user application. This file contains an include directive for user.bfm, and a procedure call to user_main (see below). Do not edit this file.

user.bfm - You can customize this file to emulate CortexM3 transactions in your system. This file contains an include directive to subsystem.bfm. The memory map of MSS and Fabric peripherals is specified inside subsystem.bfm, you can refer to those defines inside user.bfm. This file is analogous to your user application code. This file contains the procedure user_main, which is analogous to your main function in your application. You can add BFM instructions and procedures to user.bfm to emulate your application. Refer to Microsemi's DirectCore AMBA BFM User's Guide for details on the supported instructions and syntax of the BFM.

subysystem.bfm - Contains the memory map of all subsystems. You do not have to modify this file. Base addresses of AMBA slaves connected to the MSS via recognized AMBA buses in your design can be found here. This includes Fabric peripherals (connected to the MSS via the FICs) as well as MSS (Hard) peripherals.

Peripheral_init.bfm - Contains the BFM commands to initialize the MDDR/FDDR and SERDESIF. If your top level design contains any MDDR/FDDR or SERDESIF, Libero SoC automatically generates the peripherals_init.bfm file to initialize these peripherals. During simulation, the simulator executes the BFM commands in the peripherals_init.bfm before executing the user BFM commands. Do not edit this file.

The BFM files are summarized in Table 2-1.

***Table 2-1* • BFM Files**

BFM File Name	Function	Remarks
Test.bfm	• Top level BFM • Contains the main function	Libero-generated. Do not edit.
User.bfm	• Contains user BFM commands • Calls subsystem_init function	Edit this file to add user BFM commands.
Subsystem.bfm	• Contains subsystem memory map • Define name and base address of each subsystem resource • Call the init function to initialize subsystem	Do not edit.
Peripherals_init.bfm	• Contains the Memory Map of all peripherals, including SERDES • Calls the SERDES_<0/1/2/3>_init.bfm to initialize SERDES	Do not edit.

The BFM files can be accessed via the Files tab in the Simulation folder (as shown in Figure 2-2). To view the file content, double-click the file to open it in the Libero SoC Text Editor.

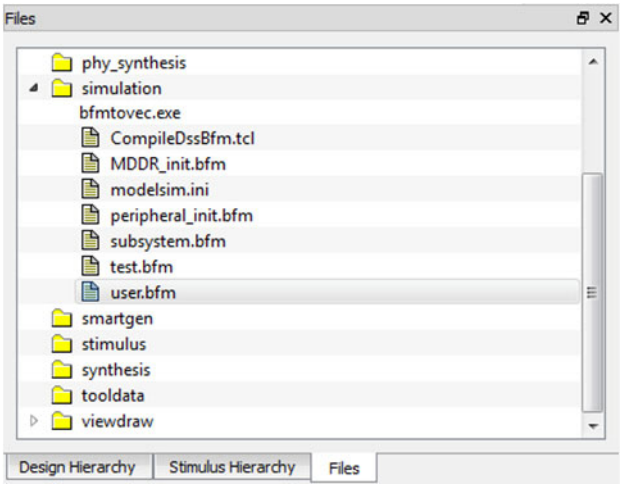

***Figure 2-2* • Simulation BFM Files**

3 – BFM Example

Writing and Verifying Fabric GPIO Bits

In the following example, two instances of CoreGPIO with 32 GPIO's per instance have been added into the Fabric. Instance CoreGPIO_0 is mapped to address 0x40055000 and Instance CoreGPIO_1 is mapped to address 0x40057000. The subsystem.bfm is automatically generated by Libero SoC and contains the memory map of the two CoreGPIO instances. CoreGPIO_0 and CoreGPIO_1 can be referenced from within your user.bfm script.

subsystem.bfm

```
#============================================================
# Created by Microsemi SmartDesign
#
# Syntax:
# -------
#
# memmap    resource_name base_address;
#
# write     width resource_name byte_offset data;
# read      width resource_name byte_offset;
# readcheck width resource_name byte_offset data;
#
#============================================================

#------------------------------------------------------------
# Memory Map
# Define name and base address of each resource.
#------------------------------------------------------------

#Peripheral Base Addresses
memmap CoreGPIO_0 0x40055000;
memmap CoreGPIO_1 0x40057000;
```

The subsystem.bfm file is generated automatically every time you generate your MSS component; you do not need to modify it.

user.bfm

To add your own BFM commands, open the user.bfm file in the Libero SoC Text Editor.

For example, in the user.bfm file below, a BFM write command is added to write the half word 0x5555 to the CoreGPIO_0 Output register. A BFM readcheck command is then added to read back from the Input Register and compared to the previously written data.

To check the syntax of your BFM commands at the end of your edits, right-click and choose **Check BFM file**. Select the Log tab to view the result of the syntax check.

```
#============================================================
# Enter your BFM commands in this file.
#.
# Syntax:
# -------
#
# memmap resource_name base_address;
#
# write width resource_name byte_offset data;
# read width resource_name byte_offset;
# readcheck width resource_name byte_offset data;
#
#============================================================
include "subsystem.bfm"
procedure user_main;
# perform subsystem initialization routine
call subsystem_init;
# add your BFM commands below:

# GPIO registers (byte wide)
# Refer Table 3-1 of CoreGPIO_HB.pdf
#Offsets from GPIO Base Address
constant INREG0 0x90
constant OUTREG0 0xA0

print "Fabric GPIO Access Start";
#Write halfword 0x5555 to CoreGPIO_0 Output Register write h CoreGPIO_0 OUTREG0 0x5555;
wait 4;
#Read halfword from CoreGPIO_0 Input Register and compare to previously written data
#(0x5555)
readcheck h CoreGPIO_0 INREG0 0x5555;
wait 4;

print "CoreGPIO_0 TEST ENDS";


wait 4;
print "CoreGPIO_1 TEST START";

#Write word 0xFFFFFFFF halfword 0x5555 to CoreGPIO_1 Output register Register
write w CoreGPIO_1 OUTREG0 0xFFFFFFFF;
wait 4;
#Read word halfword 0xFFFFFFFF from CoreGPIO_1 Input Register and compare to previously
#written data (0xFFFFFFFF )(0x5555)
readcheck w CoreGPIO_1 INREG0 0xFFFFFFFF;
wait 4;

print "CoreGPIO_1 TEST ENDS";
print "";

return
```

BFM Compiler

Libero SoC includes the BFM compiler, which converts the BFM script files into vector files (*.vec). The vector files contain a sequence of 32-bit values, each represented by an 8-digit hexadecimal value. Libero SoC is configured to automatically execute the BFM compiler when you invoke ModelSim or other supported third-party simulators from Libero SoC. Libero SoC then passes the BFM files to the Simulator for simulation (Figure 3-1). The BFM compiler also verifies the syntax of the BFM file and displays the result of the syntax check in the Log.

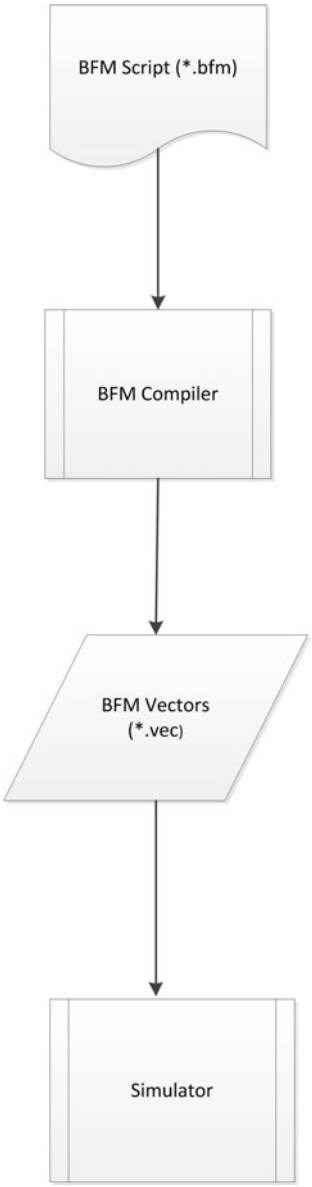

Figure 3-1 **• BFM Compiler and Simulation**

A – 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, 408.643.6913

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

Visit the Customer Support website (www.microsemi.com/soc/support/search/default.aspx) for more information and support. Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on the website.

Website

You can browse a variety of technical and non-technical information on the SoC 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. 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 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

Microsemi Corporation (NASDAQ: MSCC) offers a comprehensive portfolio of semiconductor solutions for: aerospace, defense and security; enterprise and communications; and industrial and alternative energy markets. Products include high-performance, high-reliability analog and RF devices, mixed signal and RF integrated circuits, customizable SoCs, FPGAs, and complete subsystems. Microsemi is headquartered in Aliso Viejo, Calif. Learn more at **www.microsemi.com**.

5-02-00433-2/08.14