How to use CON-FMC with Xilinx Vivado IP Integrator

Version 0 Revision 1

June 8, 2023

Future Design Systems
www.future-ds.com / contact@future-ds.com

Abstract

This document summarizes how to use CON-FMC for block design of Xilinx IP Integrator.

Table of Contents

1 Introduction

AMD/Xilinx IP Integrator provides an easy way to prepare design by instantiating and interconnecting blocks, which resides in specific repository directories.

This document explains how to integrate Future Design Systems' CON-FMC interfacing block to the IP integrator.

Followings are rough steps to use CON-FMC along with the IP Integrator. [Note that following steps are assumed to be performed on Ubuntu.]

- Step 1. Get BFM package (i.e., bfm_axi_if) for a specific FPGA board[1].
- Step 2. Add the path of the package to IP repository of your Vivado project.
- Step 3. Instantiate 'bfm_axi_if' in your block design.
- Step 4. Connect ports.

After the steps described above, the conventional Vivado design sequence follows.

2 Get BFM package

As shown in Figure 1, BFM package, i.e., bfm_axi_if resides in between host program and AMBA AXI bus, which interacts with the host program and generates read/write transactions on the system bus.

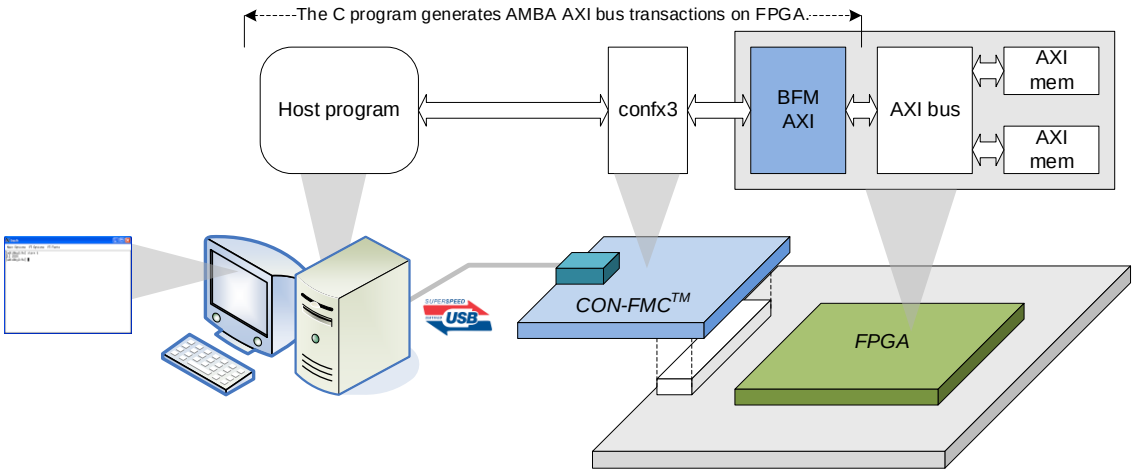

Figure 1: BFM AXI conceptual environment

[1] z7=Zynq 7000; vus=Virtex UltraScale; vusp=Virtex UltraScale+; zus=Zynq UltraScale; zusp=ZynqUltraScale+

The BFM package is a hardware block implemented in the FPGA and as a result it is FPGA dependent. The BFM package is available from Future Design Systems' home page.

- Visit http://www.future-ds.com web-page.
- Go down to 'Products' → 'CON-FMC: USB3.0 FMC Board' page.
- Select 'Expend/Collapse: Supporting FPGA boards CON-FMC'.
- Then, scroll down to find the FPGA board.
  - If not found, contact Future Design Systems.
- There will be two design resources.
  - XDC for FMC connector
  - BFM package

- Download XDC and BFM package to your local directory by clicking on it.
- Unzip and untar it by using 'tar' command and there must be 'bfm_axi_if' directory created. (Note that 'z7' should be different for your case since it stands for FPGA family.)
  - to use local directory

```
$ tar xvfz bfm_axi_if.z7.tar.gz
```

  - to use system directory, where $(CONFMC_HOME) stands for the directory where CON-FMC SW package is installed, and it will be '/opt/confmc/2020.06' by default.

```
$ sudo tar xvfz bfm_axi_if.z7.tar.gz\
    --directory $(CONFMC_HOME)/hwlib/trx_axi/lib
```

    - Now the repository directory must be '$(CONFMC_HOME)/hwlib/trx_axi/lib/bfm_axi_if'.

3 Add IP repository

Add the directory path as a new IP repository in Vivado project by clicking, Settings→IP→Repository.

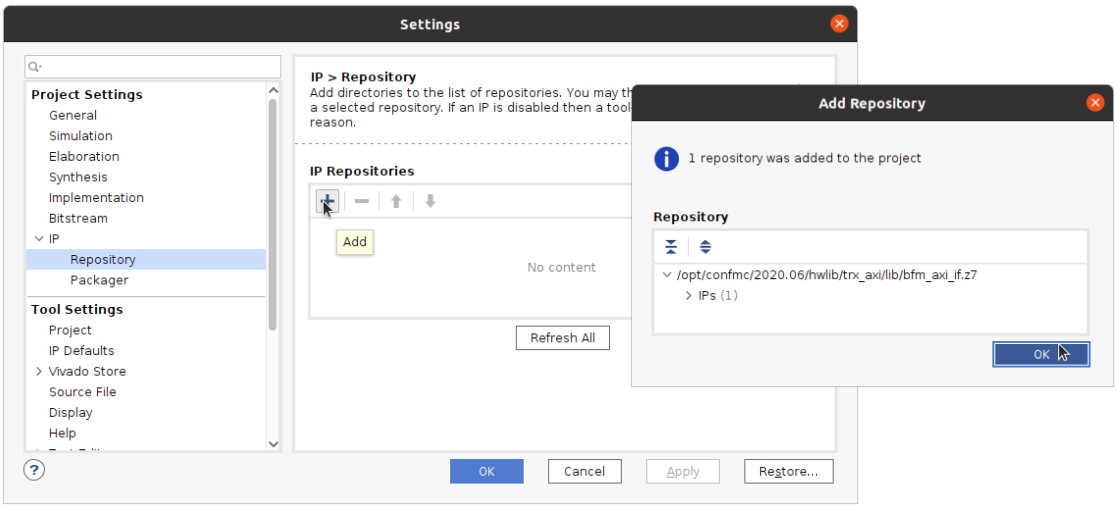

Figure 2: Add IP repository (use system directory)

4 Instantiate BFM AXI

On your block design sub-window, press right mouse button and then select ‘Add IP…’ menu. After that select ‘bfm_axi_if’ and add it to the block design.

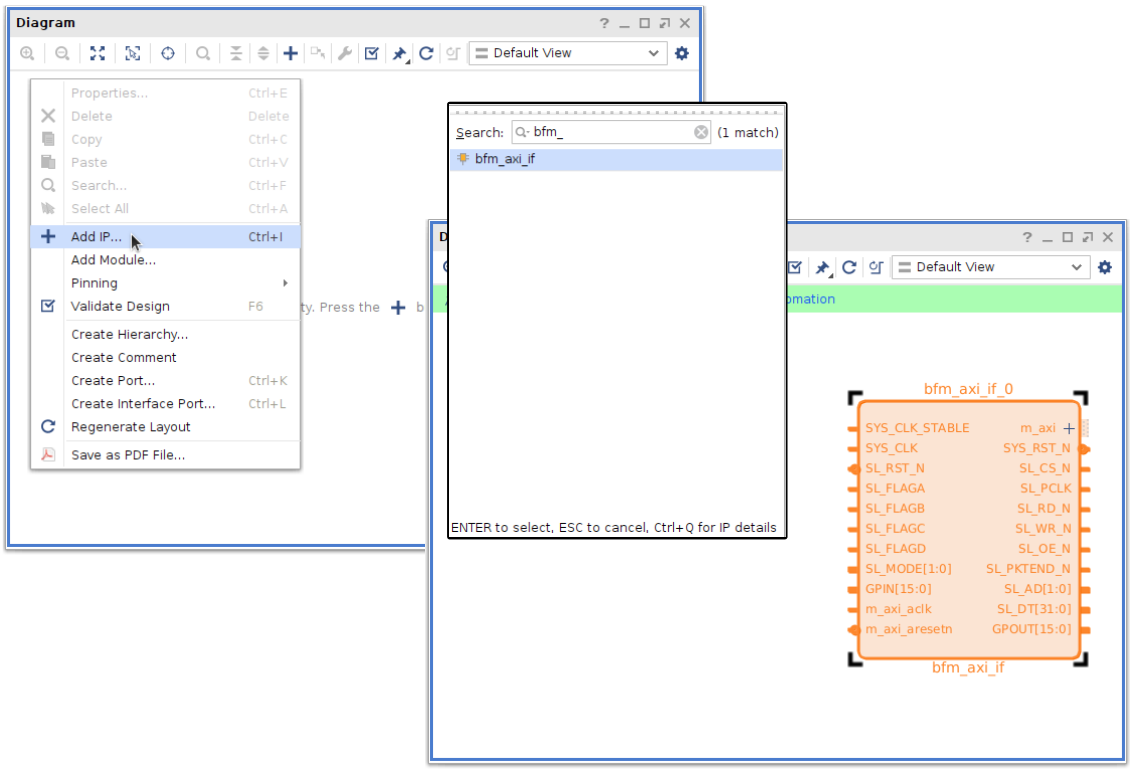

Figure 3: Instantiate BFM AXI

5 Connect ports

Figure 4 shows a typical port connection schematic, where clock and reset signals are emphasized.

- BOARD_RST_SW: active-high push-button reset input
- BOARD_CLK_IN: on-board clock input, e.g., 100Mhz
- SL_RST_N: active-low reset signal coming from CON-FMC board
- SL_PCLK: CON-FMC interfacing clock[2] going to CON-FMC board, e.g., 80Mhz derived from on-board clock input

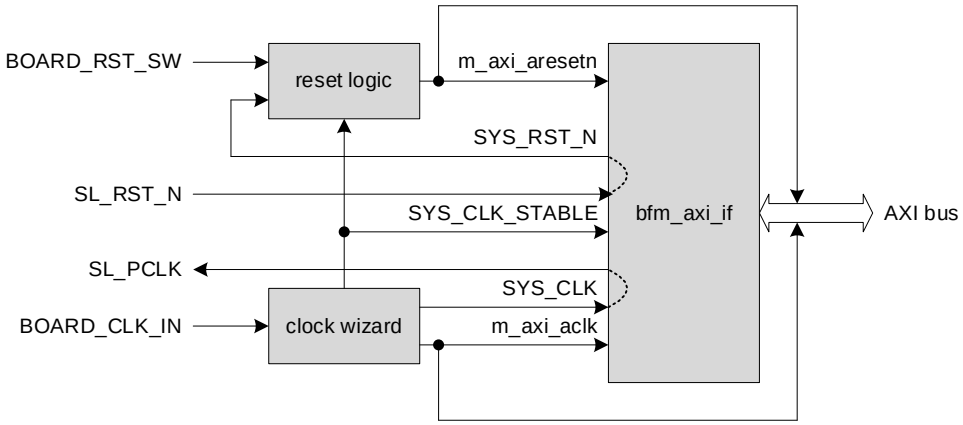

Figure 4: BFM AXI connection

Figure 5 shows an example design consisting of BFM and BRAM memory.

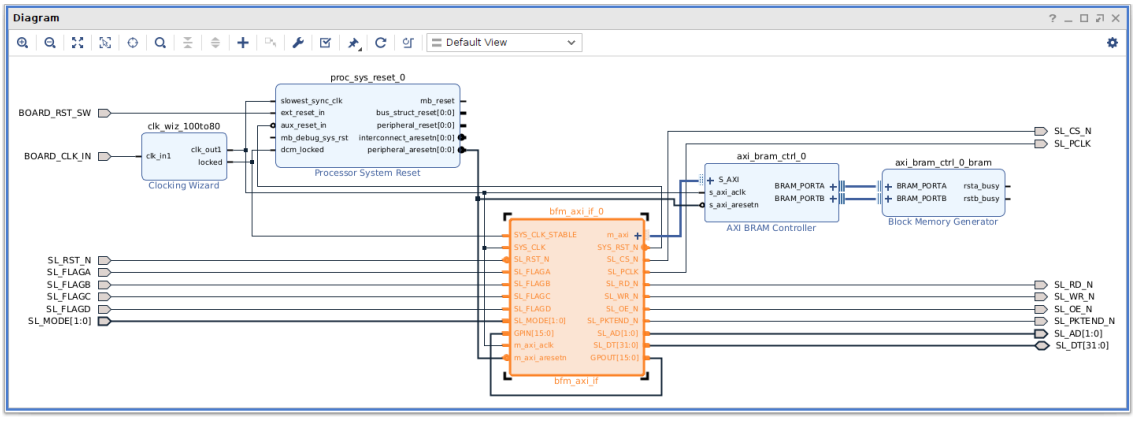

Figure 5: BFM AXI block connection example

Figure 6 shows Vivado TCL script to instantiate BFM AXI IF block and it can be called by typing ‘create_bfm_design “” in Vivado Tcl command consol.

[2] 80Mhz is recommended.

```
proc create_bfm_design { parentCell } {
  # Create instance: bfm_axi_if_0, and set properties
  set bfm_axi_if_0 [ create_bd_cell -type ip -vlnv future-ds.com:user:bfm_axi_if:1.1 bfm_axi_if_0 ]

  set SL_AD [ create_bd_port -dir O -from 1 -to 0 SL_AD ]
  set SL_CS_N [ create_bd_port -dir O SL_CS_N ]
  set SL_DT [ create_bd_port -dir IO -from 31 -to 0 -type data SL_DT ]
  set SL_FLAGA [ create_bd_port -dir I SL_FLAGA ]
  set SL_FLAGB [ create_bd_port -dir I SL_FLAGB ]
  set SL_FLAGC [ create_bd_port -dir I SL_FLAGC ]
  set SL_FLAGD [ create_bd_port -dir I SL_FLAGD ]
  set SL_MODE [ create_bd_port -dir I -from 1 -to 0 SL_MODE ]
  set SL_OE_N [ create_bd_port -dir O SL_OE_N ]
  set SL_PCLK [ create_bd_port -dir O -type clk SL_PCLK ]
  set SL_PKTEND_N [ create_bd_port -dir O SL_PKTEND_N ]
  set SL_RD_N [ create_bd_port -dir O SL_RD_N ]
  set SL_RST_N [ create_bd_port -dir I -type rst SL_RST_N ]
  set SL_WR_N [ create_bd_port -dir O SL_WR_N ]
  connect_bd_net -net Net [get_bd_ports SL_DT] [get_bd_pins bfm_axi_if_0/SL_DT]
  connect_bd_net -net SL_FLAGA_0_1 [get_bd_ports SL_FLAGA] [get_bd_pins bfm_axi_if_0/SL_FLAGA]
  connect_bd_net -net SL_FLAGB_0_1 [get_bd_ports SL_FLAGB] [get_bd_pins bfm_axi_if_0/SL_FLAGB]
  connect_bd_net -net SL_FLAGC_0_1 [get_bd_ports SL_FLAGC] [get_bd_pins bfm_axi_if_0/SL_FLAGC]
  connect_bd_net -net SL_FLAGD_0_1 [get_bd_ports SL_FLAGD] [get_bd_pins bfm_axi_if_0/SL_FLAGD]
  connect_bd_net -net SL_MODE_0_1 [get_bd_ports SL_MODE] [get_bd_pins bfm_axi_if_0/SL_MODE]
  connect_bd_net -net SL_RST_N_0_1 [get_bd_ports SL_RST_N] [get_bd_pins bfm_axi_if_0/SL_RST_N]
  connect_bd_net -net bfm_axi_if_0_SL_AD [get_bd_ports SL_AD] [get_bd_pins bfm_axi_if_0/SL_AD]
  connect_bd_net -net bfm_axi_if_0_SL_CS_N [get_bd_ports SL_CS_N] [get_bd_pins bfm_axi_if_0/SL_CS_N]
  connect_bd_net -net bfm_axi_if_0_SL_OE_N [get_bd_ports SL_OE_N] [get_bd_pins bfm_axi_if_0/SL_OE_N]
  connect_bd_net -net bfm_axi_if_0_SL_PCLK [get_bd_ports SL_PCLK] [get_bd_pins bfm_axi_if_0/SL_PCLK]
  connect_bd_net -net bfm_axi_if_0_SL_PKTEND_N [get_bd_ports SL_PKTEND_N] [get_bd_pins
bfm_axi_if_0/SL_PKTEND_N]
  connect_bd_net -net bfm_axi_if_0_SL_RD_N [get_bd_ports SL_RD_N] [get_bd_pins bfm_axi_if_0/SL_RD_N]
  connect_bd_net -net bfm_axi_if_0_SL_WR_N [get_bd_ports SL_WR_N] [get_bd_pins bfm_axi_if_0/SL_WR_N]
}
```

Figure 6: BFM AXI block port connection script

6 References

[1] AMD/Xilinx, Vivado Design Suite User Guide: Designing IP Subsystems Using IP Integrator, UG994.
[2] Future Design Systems, CON-FMC User Manual, FDS-TD-2018-03-001.
[3] Future Design Systems, TRX_AXI: BMBA AXI Transactor for GPIF2MST, FDS-TD-2018-04-008.

7 Revision history

☐ 2023.6.8: Document started by Ando Ki.

– End of document –