eMMC Host Controller 5.1 IP Integration Manual

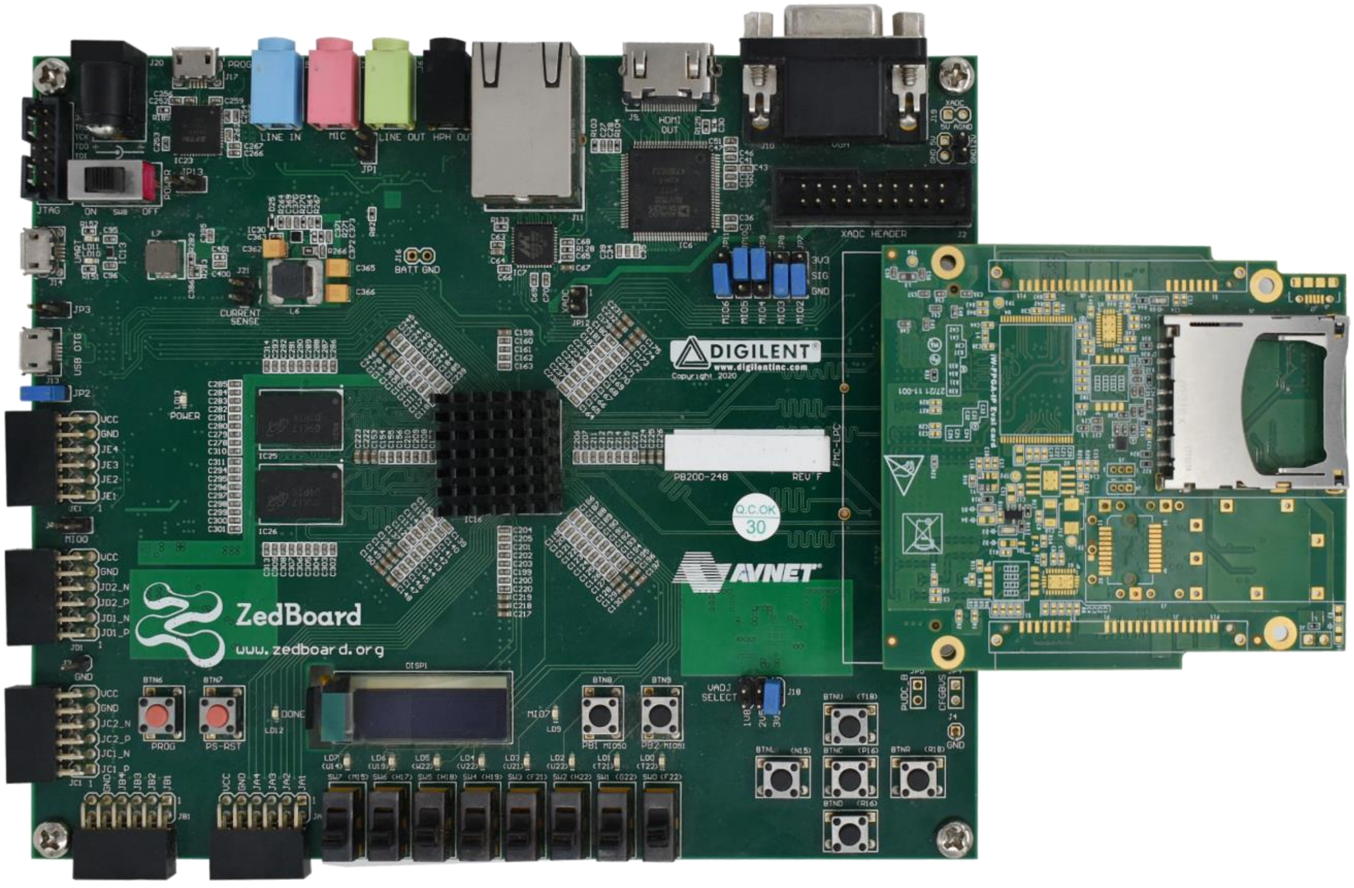


Table of Content

List Of Figures

List Of Tables

1 Introduction

1.1 Purpose

The purpose of this document is to describe eMMC Host Controller 5.1 IP Integration details.

1.2 Overview

eMMC Host Controller interfaces the ZYNQ-7 Processor through AXI4 bus enabling the data transfers between each other. The ZYNQ-7 Processor will send the response to the eMMC host depending on the command issued.

1.3 Acronyms and Abbreviations

Table 1: Acronyms & Abbreviations

Term	Meaning
FPGA	Field Programmable Gate Array
GPIO	General purpose input output
FMC	FPGA Mezzanine Card
LUT	Look Up Table
IO	Input and Output
FF	Flip Flop

2 eMMC Host Controller 5.1

The below figure represents the test setup for the ZED development Board with eMMC FMC Card.

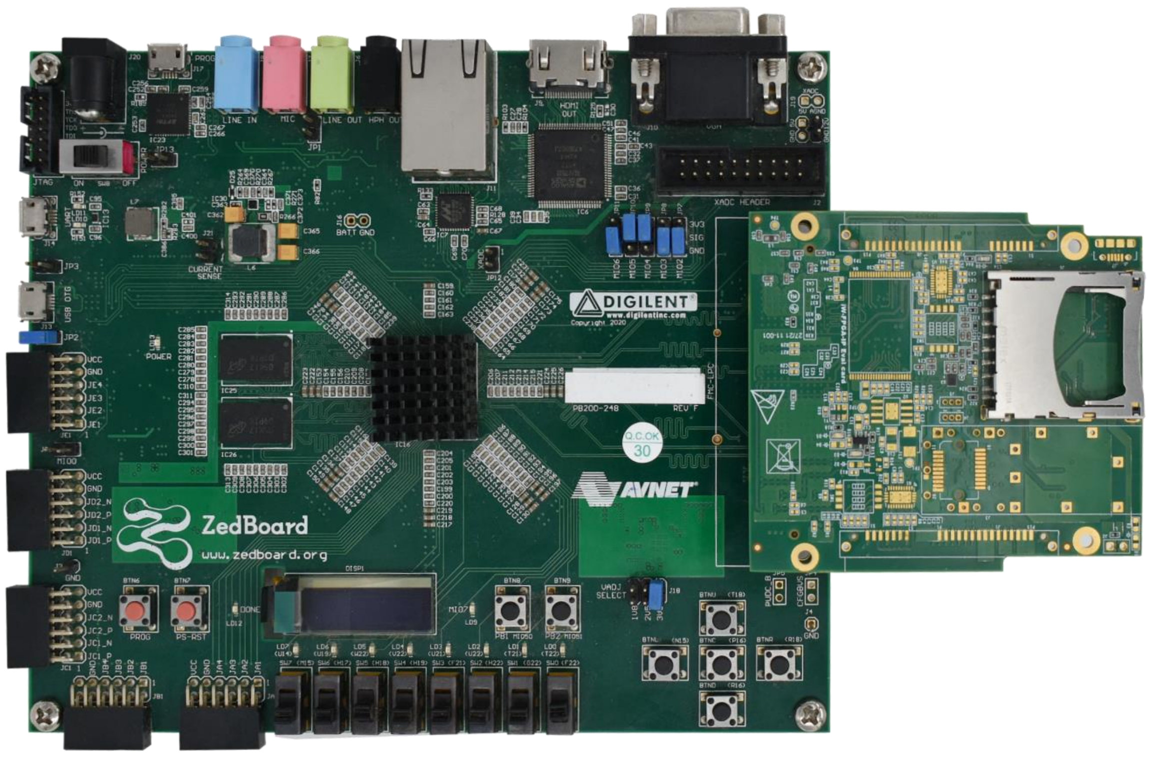
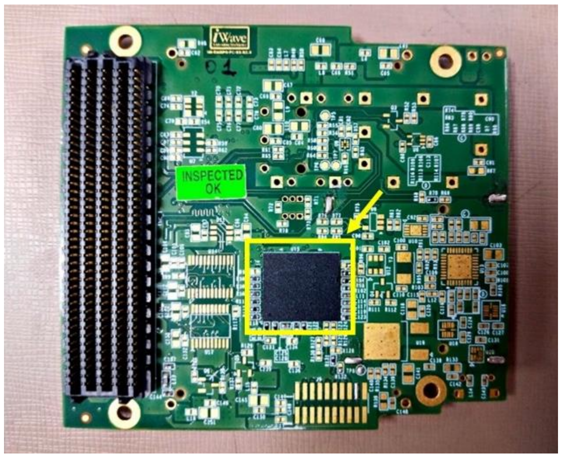


Figure 1: eMMC Host Controller 5.1 setup

2.1 eMMC FMC Card

Figure 2: eMMC FMC Card



3 IP Configuration and Instantiation

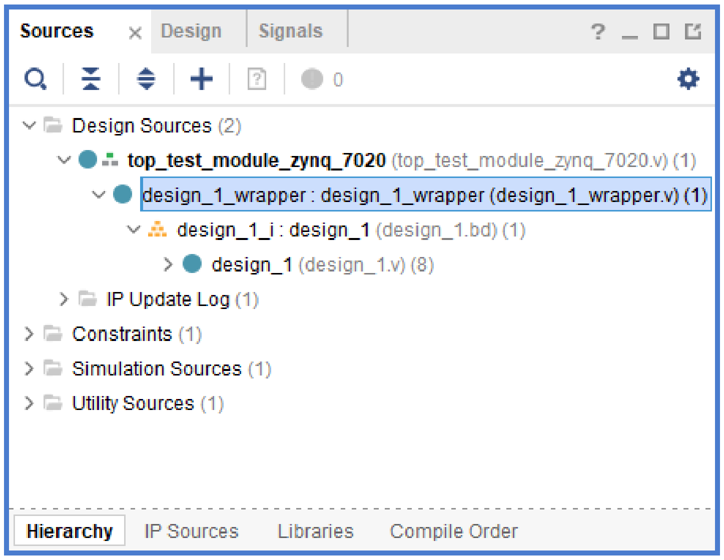
3.1 Example design

The eMMC host controller 5.1 IP example design mainly consists of

1. **ZYNQ7 processor:** Zynq-7 Processor is used to configure eMMC host controller IP mainly access to register set and to read/write to DDR memory
2. **eMMC Host Controller 5.1 IP:** eMMC host controller 5.1 IP takes the command given from the Processor to the register set and with the ADMA the read and write operation will happen from and to the eMMC Card.

3.2 eMMC Host Ccontroller 5.1 IP Instantiation

The eMMC Host Controller block design (design_1_wrapper.v) is instantiated in the design as shown in the below Figure 3.



Design Sources (2)
- top_test_module_zynq_7020 (top_test_module_zynq_7020.v) (1)
  - design_1_wrapper : design_1_wrapper (design_1_wrapper.v) (1)
    - design_1_i : design_1 (design_1.bd) (1)
      - design_1 (design_1.v) (8)
  - IP Update Log (1)

Constraints (1)
Simulation Sources (1)
Utility Sources (1)

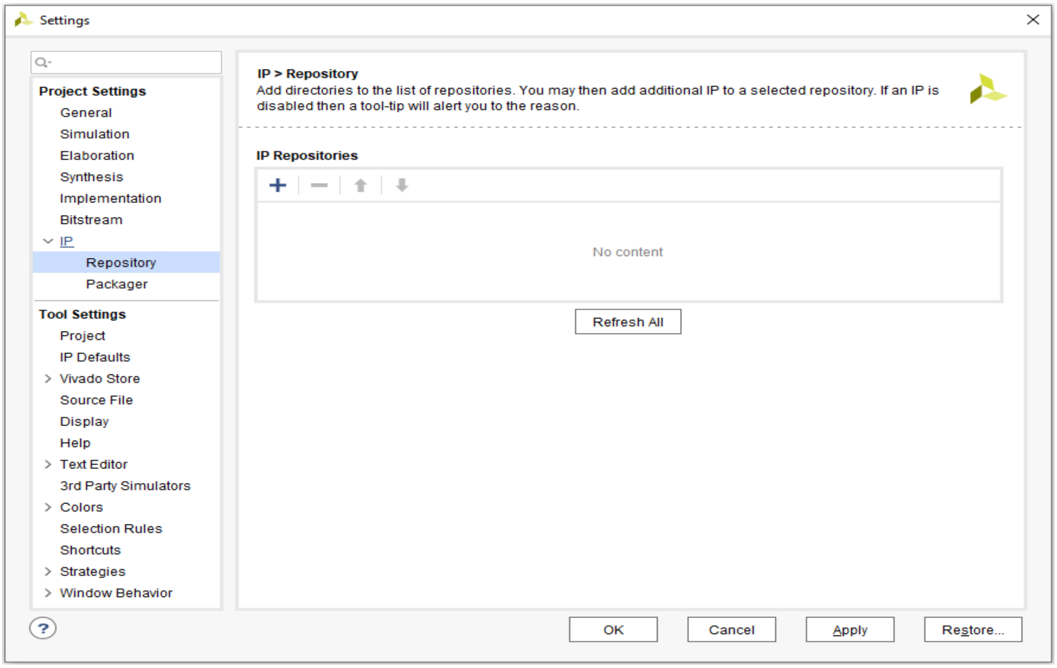
Figure 3: Instantiation Module of eMMC Host Controller 5.1 IP

Module top_test_module_zynq_7020 is the top module of the project which integrates the wrapper files to form a single IP.

3.3 Steps to Configuring the eMMC Host Controller 5.1 IP

- Install the required Vivado Design Suite 2021.2 for the host PC adding the license path. Downloads (xilinx.com)
- Open the Xilinx Vivado tool and click on “Open Project” and select the required project.
- Now the project selected will be opened in Vivado Design.

- So once the IP is generated then make sure that the IP need to be added to the project so copy the IP folder from the path that was saved to the local to project folder and go to settings of Vivado and add the IP to the Project Settings → IP → Repository→then add the IP path.



IP > Repository

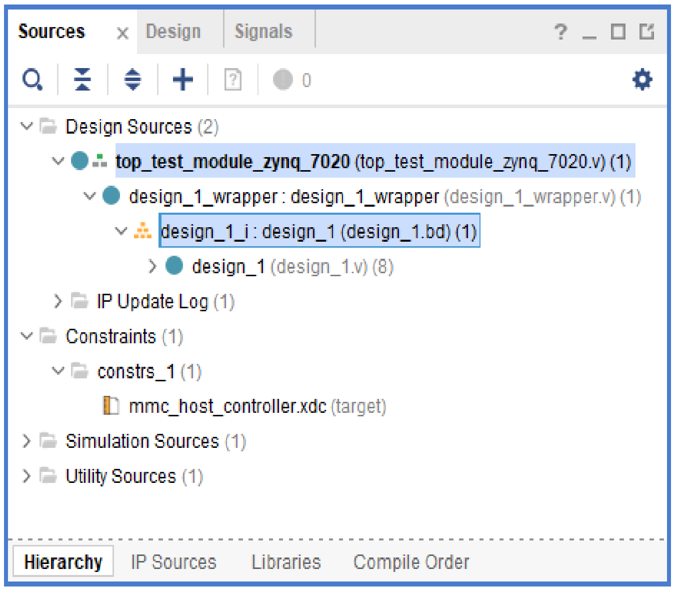
Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.

IP Repositories

Figure 4: eMMC Host Controller 5.1 IP Configuration step 1

- IP location is ...\E:\eMMC\eMMC_ZED_board_2021_2_CQE\zed_emmc_cqe\project _1.srcs\sources_1\bd\design_1\IP. Add IP by clicking on the '+' sign as shown in Figure 4 above. After the IP is added click on Apply and close the dialogue box.

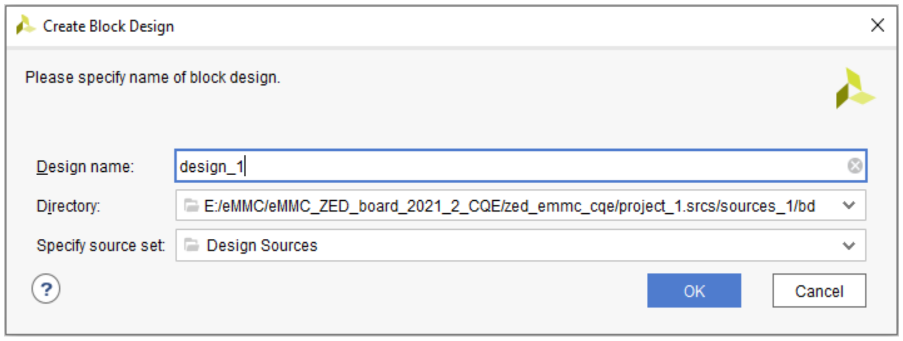
- Once the IP is added in the project repository open the block design and add the other IP's like Zynq-7 processor IP, processor system reset, AXI4-interconnect and Clocking Wizard, add the "sd_host_controller_v1_3" IP to the block design by clicking on deisgn_1 under top_test_module_zynq_7020 → design_1_wrapper present in the source tab.



Design Sources (2)
top_test_module_zynq_7020 (top_test_module_zynq_7020.v) (1)
design_1_wrapper : design_1_wrapper (design_1_wrapper.v) (1)
design_1_i : design_1 (design_1.bd) (1)
design_1 (design_1.v) (8)
IP Update Log (1)
Constraints (1)
constrs_1 (1)
mmc_host_controller.xdc (target)
Simulation Sources (1)
Utility Sources (1)

Figure 5: eMMC Host Controller 5.1 IP Configuration step 2

- How the block design was created was showed in the below Figures. Flow navigator → Project Manager → IP Integrator → open "Create Block design".

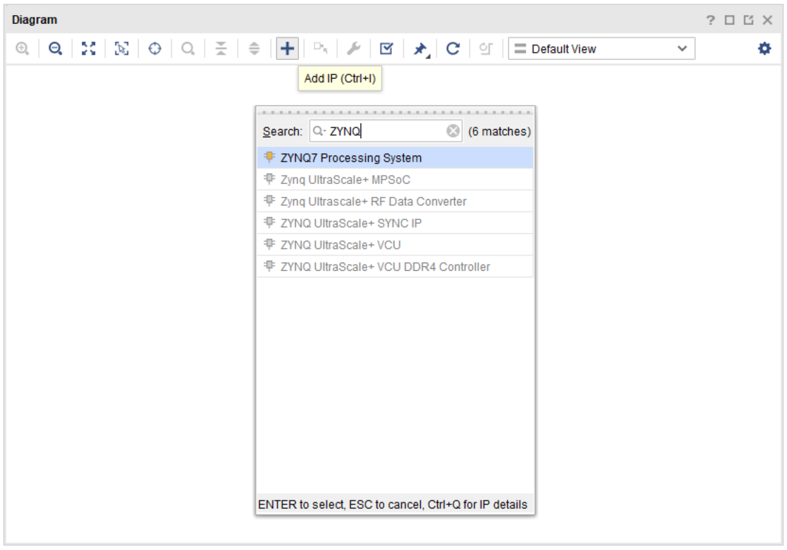


Please specify name of block design.

Design name: design_1
Directory: E:/eMMC/eMMC_ZED_board_2021_2_CQE/zed_emmc_cqe/project_1.srcs/sources_1/bd
Specify source set: Design Sources

Figure 6: eMMC Host Controller 5.1 IP Configuration step 3

- Diagram, Address Editor and Address Map windows will be opened. In Diagram window, click '+' to add the IP into the block deign. What is the required IP need for the design that should be added.



Search: ZYNQ (6 matches)

ZYNQ7 Processing System
Zynq UltraScale+ MPSoC
Zynq Ultrascale+ RF Data Converter
ZYNQ UltraScale+ SYNC IP
ZYNQ UltraScale+ VCU
ZYNQ UltraScale+ VCU DDR4 Controller

ENTER to select, ESC to cancel, Ctrl+Q for IP details

Figure 7: eMMC Host Controller 5.1 IP Configuration step 4

- After adding the other IPs like Zynq-7 processor IP, processor system reset, AXI4-interconnect and Clocking Wizard, add the **sd_host_controller_v1_3** IP to the block design by clicking on the '+' sign as shown in the below Figure 8. The **sd_host_controller_v1_3** IP was configured as the eMMC Host Controller 5.1 IP of the design.

  Note: The Xilinx ZED Development board example design doesn't have any eMMC Host Controller IP, so the sd_host_controller_v1_3 IP was configured as an eMMC Host Controller IP. It was used in the block design after being configured.

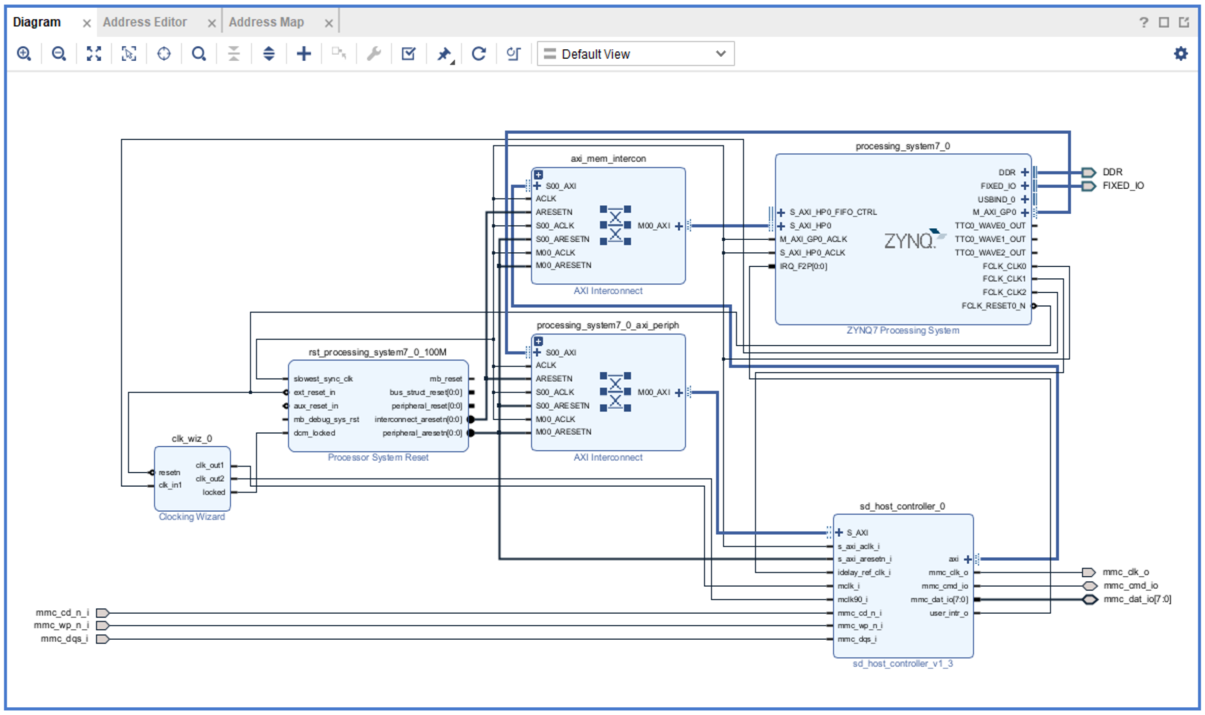
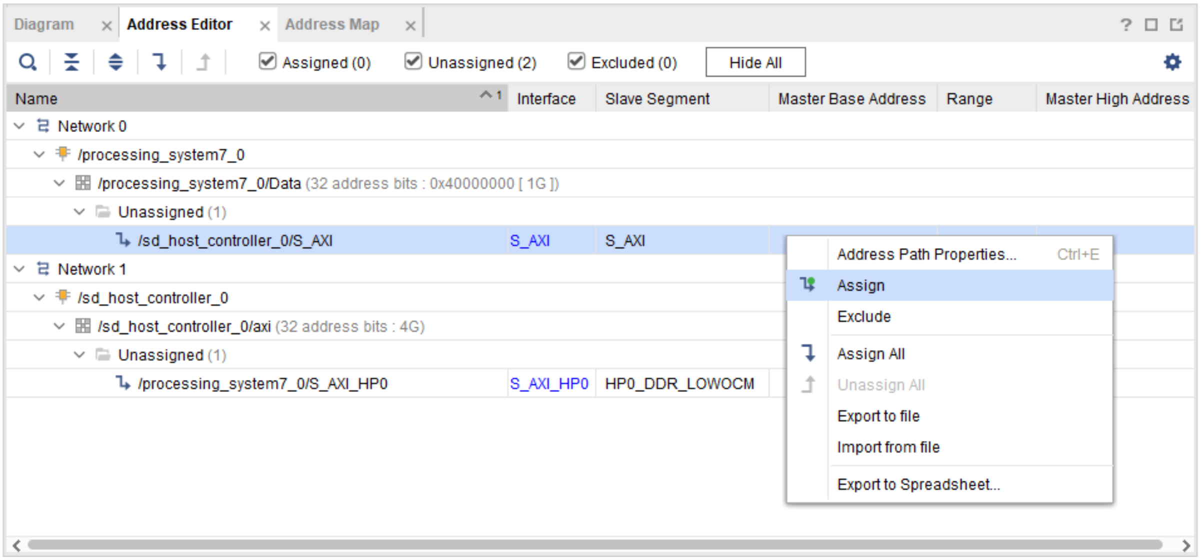


Figure 8: eMMC Host Controller 5.1 IP Configuration step 5

- Once the block design was created, then go to Address Editor window to assign the address to Zynq-7 processor and sd_host_controller IP as shown in Figure 9.



Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/processing_system7_0					
/processing_system7_0/Data (32 address bits : 0x40000000 [1G])					
Unassigned (1)					
/sd_host_controller_0/S_AXI	S_AXI	S_AXI			
Network 1					
/sd_host_controller_0					
/sd_host_controller_0/axi (32 address bits : 4G)					
Unassigned (1)					
/processing_system7_0/S_AXI_HP0	S_AXI_HP0	HP0_DDR_LOWOCM			

Figure 9: eMMC Host Controller 5.1 IP Configuration step 6

- The addresses were showed in the Address Map window After being assigned the address to Zynq-7 processor and sd_host_controller IP. Address of Zynq-7 processor as **0x43c0_0000** and sd_host_controller as **0x0.**

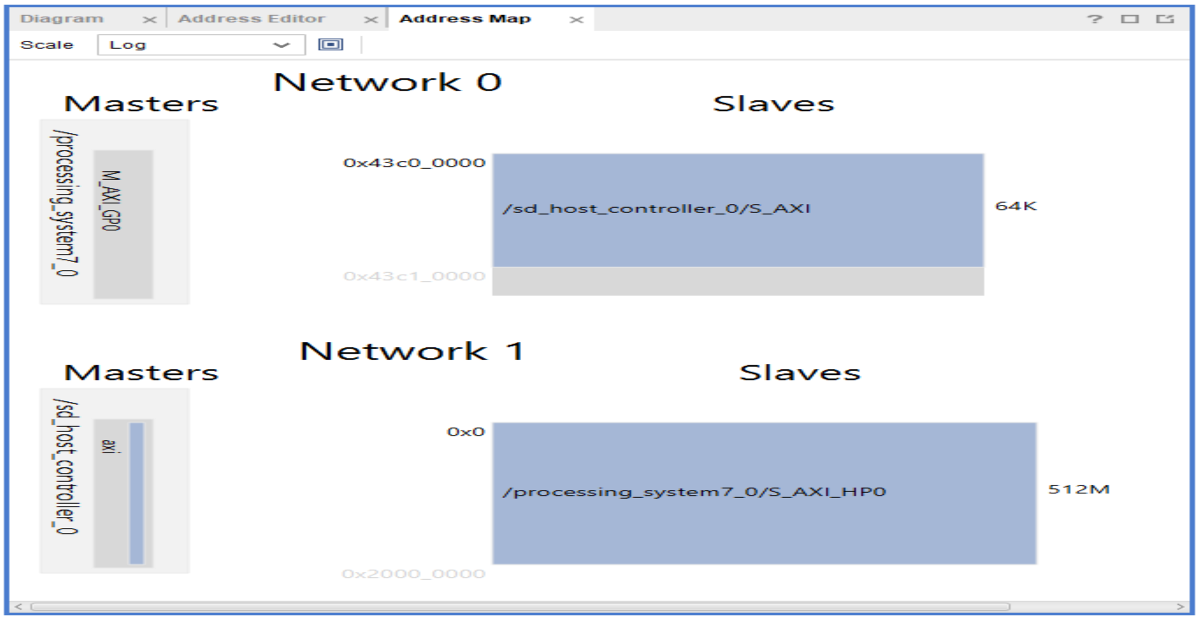


Figure 10: eMMC Host Controller 5.1 IP Configuration step 7

- Synthesize the block design first by validating the design (press key 'F6') and generate the outputs. Once the design validation was completed then generate wrapper module of the block design by right click the .bd file and click Create HDL wrapper as shown in Figure 11. After the wrapper file creation then Instantiate inside the top module of design as shown in Figure 3.

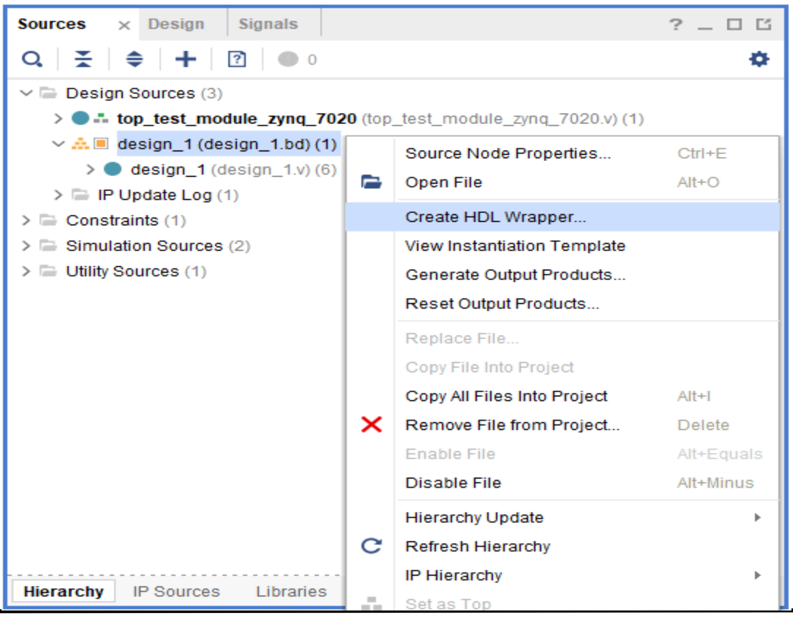
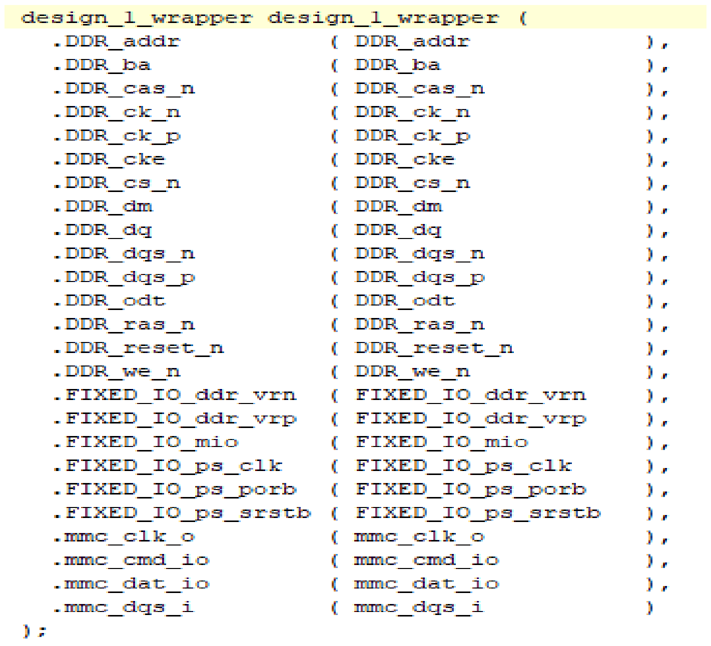


Figure 11: eMMC Host Controller 5.1 IP Configuration step 8

- Instantiate design_1_wrapper in the top module with the ports and signals of block design, in the top_test_module_zynq_7020.v file as shown in the Figure 12.



```
design_1_wrapper design_1_wrapper (
    .DDR_addr             ( DDR_addr              ),
    .DDR_ba               ( DDR_ba                ),
    .DDR_cas_n            ( DDR_cas_n             ),
    .DDR_ck_n             ( DDR_ck_n              ),
    .DDR_ck_p             ( DDR_ck_p              ),
    .DDR_cke              ( DDR_cke               ),
    .DDR_cs_n             ( DDR_cs_n              ),
    .DDR_dm               ( DDR_dm                ),
    .DDR_dq               ( DDR_dq                ),
    .DDR_dqs_n            ( DDR_dqs_n             ),
    .DDR_dqs_p            ( DDR_dqs_p             ),
    .DDR_odt              ( DDR_odt               ),
    .DDR_ras_n            ( DDR_ras_n             ),
    .DDR_reset_n          ( DDR_reset_n           ),
    .DDR_we_n             ( DDR_we_n              ),
    .FIXED_IO_ddr_vrn     ( FIXED_IO_ddr_vrn      ),
    .FIXED_IO_ddr_vrp     ( FIXED_IO_ddr_vrp      ),
    .FIXED_IO_mio         ( FIXED_IO_mio          ),
    .FIXED_IO_ps_clk      ( FIXED_IO_ps_clk       ),
    .FIXED_IO_ps_porb     ( FIXED_IO_ps_porb      ),
    .FIXED_IO_ps_srstb    ( FIXED_IO_ps_srstb     ),
    .mmc_clk_o            ( mmc_clk_o             ),
    .mmc_cmd_io           ( mmc_cmd_io            ),
    .mmc_dat_io           ( mmc_dat_io            ),
    .mmc_dqs_i            ( mmc_dqs_i             )
);
```

Figure 12: Instantiation of eMMC Host Controller 5.1 IP

- The Constraint file (.xdc) provided in the design is for ZED Development Board and should be changed for custom boards
- Give the required clock, Pin/IO constraints for eMMC Host Controller 5.1 in the .xdc file and compile the custom design with eMMC Host 5.1 IP.

4 Implementation Details

4.1 Clock Domain

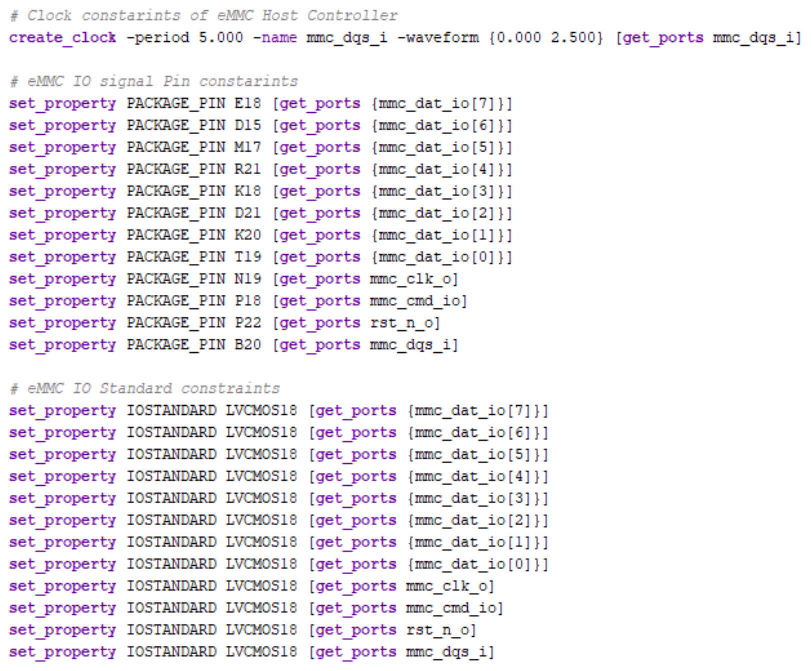
eMMC Host Controller 5.1 IP works on the HS400_ES mode and the base clock mclk_i (phase shifted 60 degree) drives the output clock to the eMMC card. User may need to adjust this clock between 100 MHz to 200 MHz based on the hardware setup. In other words, if user gets CRC error with say 200 MHz, then user can reduce the clock to say 190 MHz or below than to make sure write and read works fine with that base clock.

mclk90_i is a phase shifted version of the mclk_i which is 200MHz and this clock is phase shifted based on the response from the eMMC card. Idelay_ref_clk_i is the fixed 200 MHz clock which will be provided for the IDELAY primitive to be used for the tuning implementation block. s_axi_clk_i which is 100 MHz which will be used for the communication between the processor and IP using the AXI4-lite interface for register access and AXI4-MM for DMA access.

mclk_i (200Mhz) and mclk90_i (200Mhz) clocks were generated by the clocking wizard in the block design.

4.2 Constraints

Figure 13 shows the pin constraints in the .xdc file of example design.



```
# Clock constarints of eMMC Host Controller
create_clock -period 5.000 -name mmc_dqs_i -waveform {0.000 2.500} [get_ports mmc_dqs_i]

# eMMC IO signal Pin constarints
set_property PACKAGE_PIN E18 [get_ports {mmc_dat_io[7]}]
set_property PACKAGE_PIN D15 [get_ports {mmc_dat_io[6]}]
set_property PACKAGE_PIN M17 [get_ports {mmc_dat_io[5]}]
set_property PACKAGE_PIN R21 [get_ports {mmc_dat_io[4]}]
set_property PACKAGE_PIN K18 [get_ports {mmc_dat_io[3]}]
set_property PACKAGE_PIN D21 [get_ports {mmc_dat_io[2]}]
set_property PACKAGE_PIN K20 [get_ports {mmc_dat_io[1]}]
set_property PACKAGE_PIN T19 [get_ports {mmc_dat_io[0]}]
set_property PACKAGE_PIN N19 [get_ports mmc_clk_o]
set_property PACKAGE_PIN P18 [get_ports mmc_cmd_io]
set_property PACKAGE_PIN P22 [get_ports rst_n_o]
set_property PACKAGE_PIN B20 [get_ports mmc_dqs_i]

# eMMC IO Standard constraints
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[7]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[6]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[5]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[4]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[3]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[2]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[1]}]
set_property IOSTANDARD LVCMOS18 [get_ports {mmc_dat_io[0]}]
set_property IOSTANDARD LVCMOS18 [get_ports mmc_clk_o]
set_property IOSTANDARD LVCMOS18 [get_ports mmc_cmd_io]
set_property IOSTANDARD LVCMOS18 [get_ports rst_n_o]
set_property IOSTANDARD LVCMOS18 [get_ports mmc_dqs_i]
```

Figure 13: Constraints of eMMC Host Controller 5.1 IP

NOTE: These constraints are in accordance to example design created for the ZED Development kit and there is no need of adding the MIG/ DDR IO, UART IO, clock and reset pins. Define the constraints for clock, reset and UART pins accordingly for any other custom board.

5 Design modification to be done for Custom Board

- Update the FPGA part number/board according to the FPGA device used
- Update the complete design for the selected FPGA device
- Set mmc_clk and mmc90_clk frequency to 200MHz
- Update the pin constraints for eMMC interface, clock, reset
- Update the clock constraint according to the input clock frequency for the selected FPGA device
- Recompile the design to generate the new binaries and use the XSA file to create the new application project in Vitis

6 Resource Utilization

The table below shows the resource utilization summary for ZED development Board for eMMC Host Controller IP.

Table 2: Resource Utilization for device for ZED development Board.

Resource	Utilization	Available
LUT	6117	53200
LUTRAM	237	17400
FF	7559	106400
BRAM	16	140