SD 3.0 Host Integration Manual

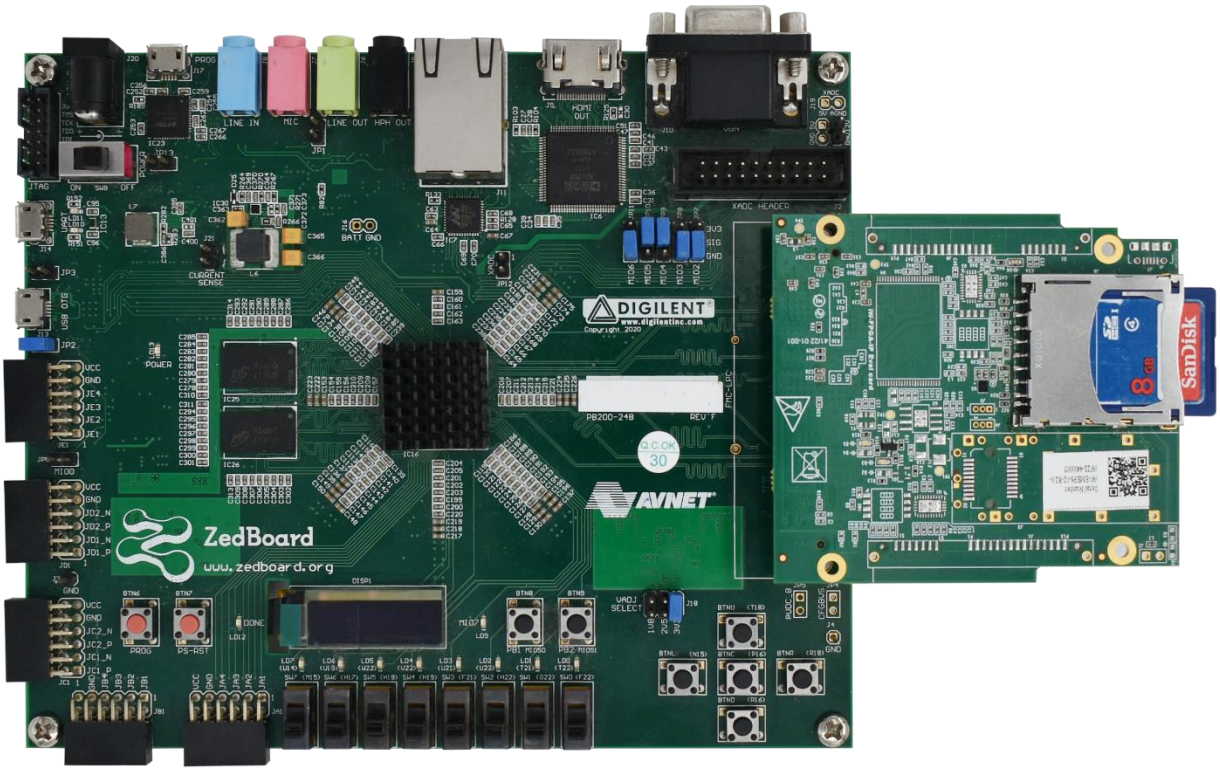

Table of Content

List Of Figures

List Of Tables

1 Introduction

1.1 Purpose

The purpose of this document is to describe SD Host Controller integration details.

1.2 Scope

This document describes the SD Host Controller. This document contains information about IPintegration details.

1.3 Reference Document

- SD Physical Specification Version 3.00
- SD Host Controller design document (iW-EMEXU-DD-01-R1.0-REL1.0.doc).

1.4 Overview

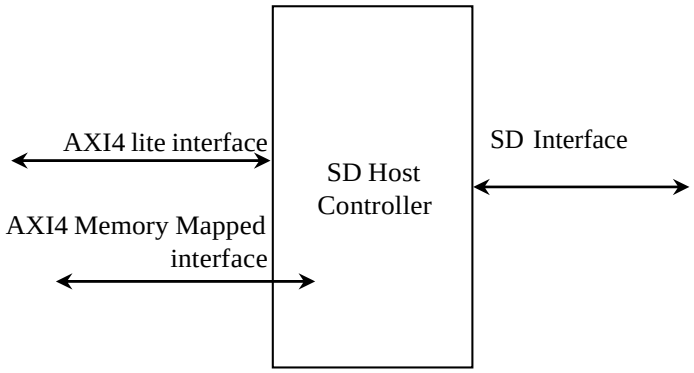

Figure 1: SD Host Controller Block Diagram

SD Host Controller defines a standard register set to control SD memory card. iWave SuppliesSD Host controller as IP catalog component.

1.5 Acronyms and Abbreviations

Table 1: Acronyms & Abbreviations

Term	Meaning
FPGA	Field Programmable Gate Array
RTL	Register Transfer Logic
SD	Secure Digital

2 IP Instantiation

2.1 Step to instantiate the iWave SD Host Controller IP

Below steps are given to add iWave SD Host controller IP to a existing Vivado Block designproject.

Below figure shows the high-level block diagram of project created targeting Zed Board ZynqEvaluation and Development Kit.

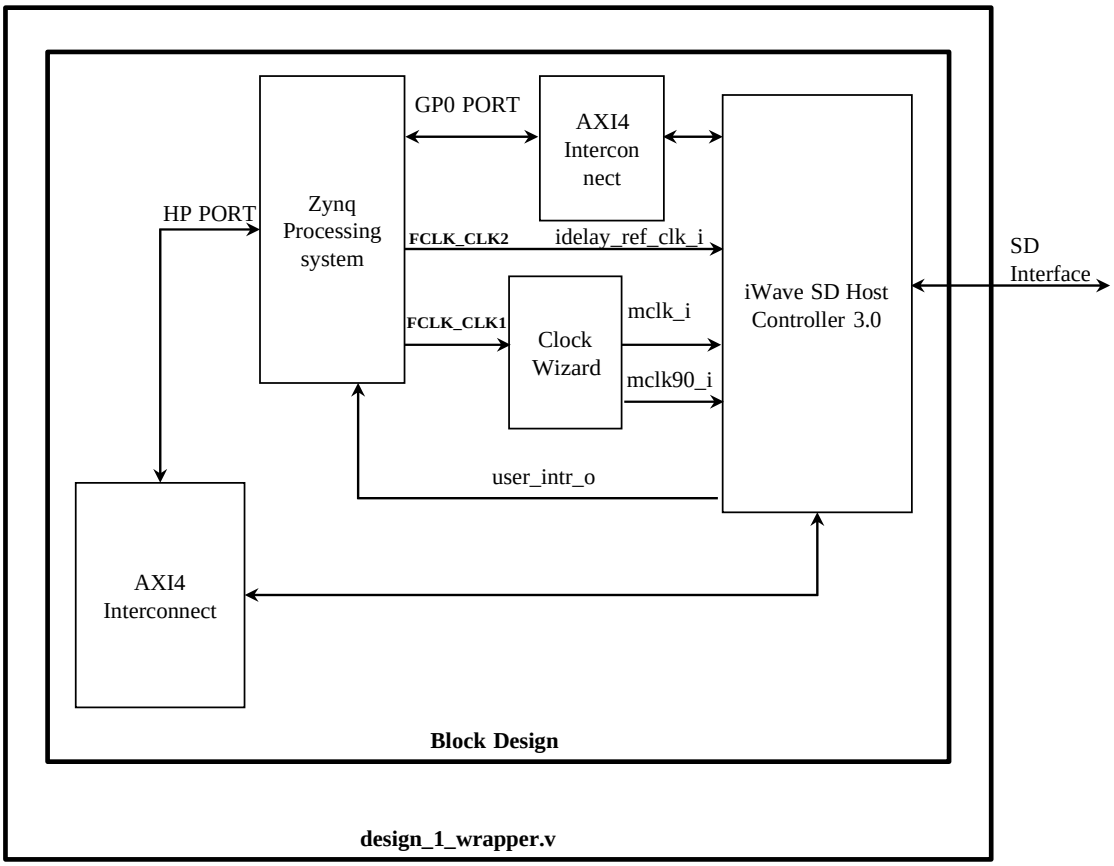

Figure 2: Example design block diagram

Following steps explains the procedure to add the iWave SD Host Controller IP.Copy the Release package and paste at say E:\

1. Copy the content in the Folder E:\iW-EMEXU-FF-R1.0-REL1.4\iW-EMEXU-FS-R1.0-REL1.0 to E:\iW-EMEXU-FF-R1.0-REL1.4\iW-EMEXU-ED-R1.0-REL1.0\project_1

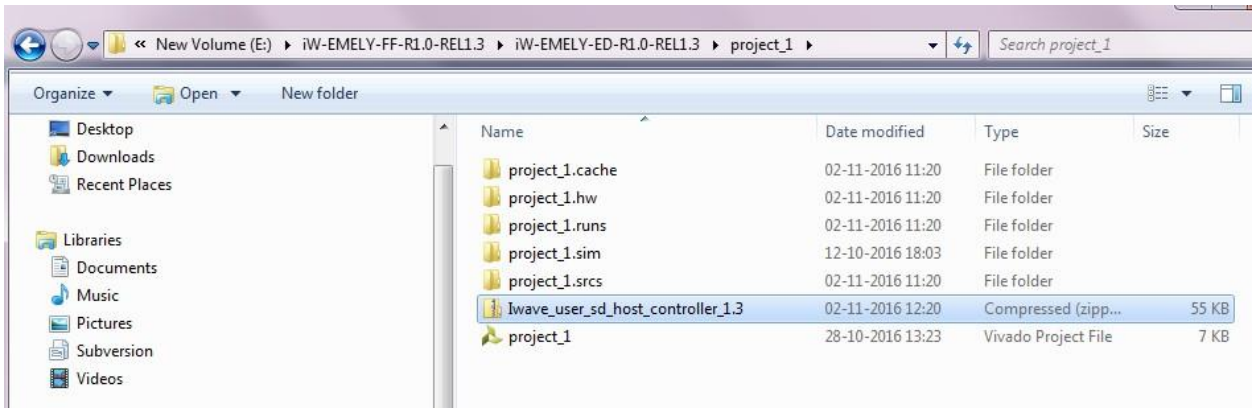

Figure 3: Step#1 to instantiate the iWave SD Host Controller IP in Design

2. Open the project and goto Tools -> Project Settings

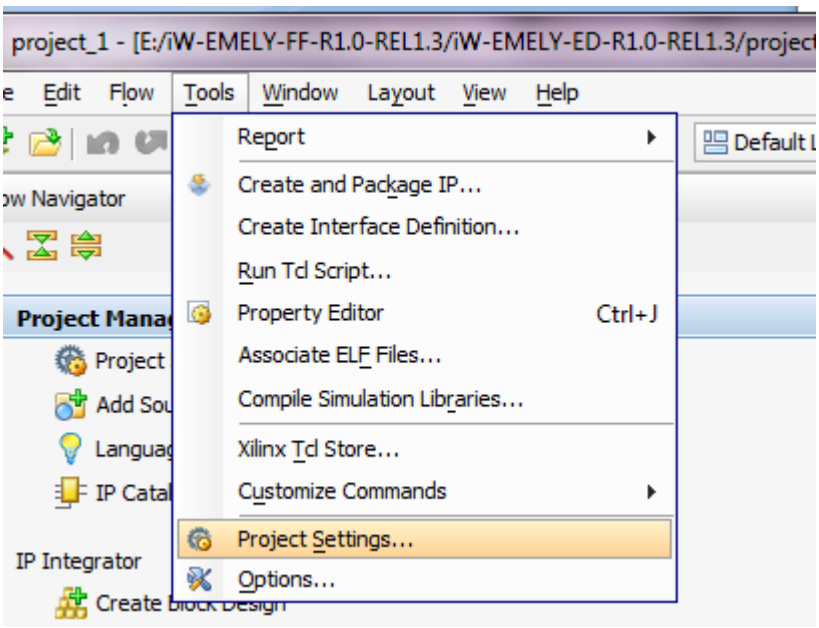

Figure 4: Step#2 to instantiate the iWave SD Host Controller IP in Design

3. In Project setting go to IP, as shown below

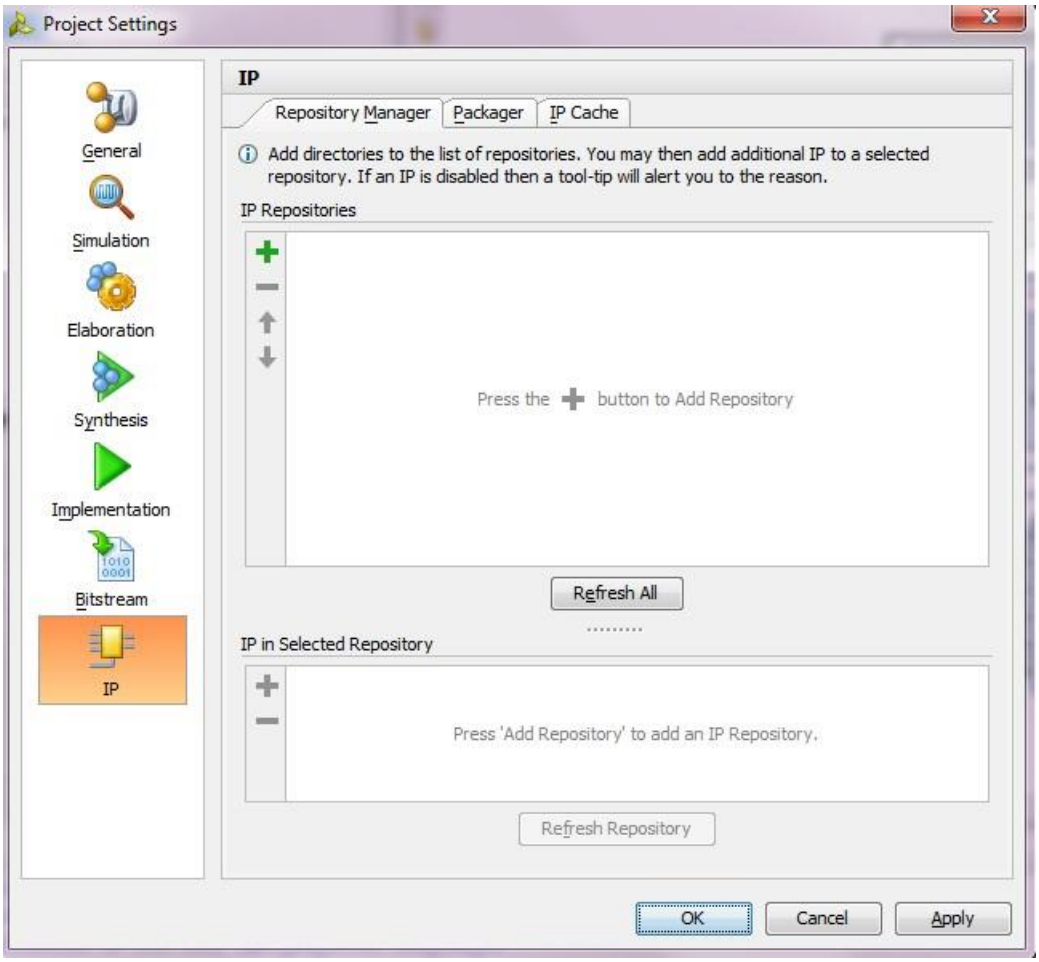

Figure 5: Step#3 to instantiate the iWave SD Host Controller IP in Design

4.Click on + in IP Repositories and set it to E:\iW-EMEXU-FF-R1.0-REL1.4\iW-EMEXU-ED-R1.0-REL1.0\project_1
and Click on + in Selected Repository and add ***Iwave_user_sd_host_controller_1.3.zip*** IP andClick OK as shown below

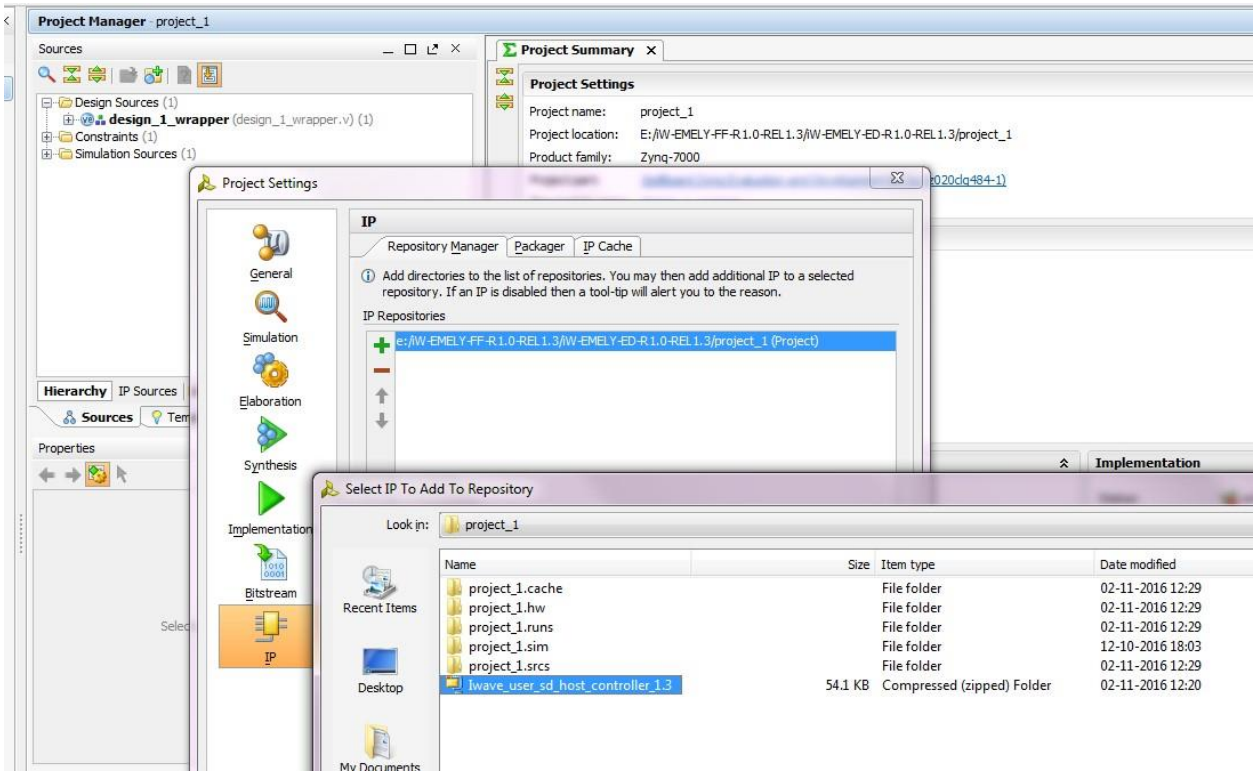

Figure 6: Step#4 to instantiate the iWave SD Host Controller IP in Design

5.Open the Block design where the IP need to be added and right click and select "Add IP"option in the IP catalog search for iWave IP as shown below.

Figure 7: Step#5 to instantiate the iWave SD Host Controller IP in Design

6. Add iWave SD Host Controller IP as shown below

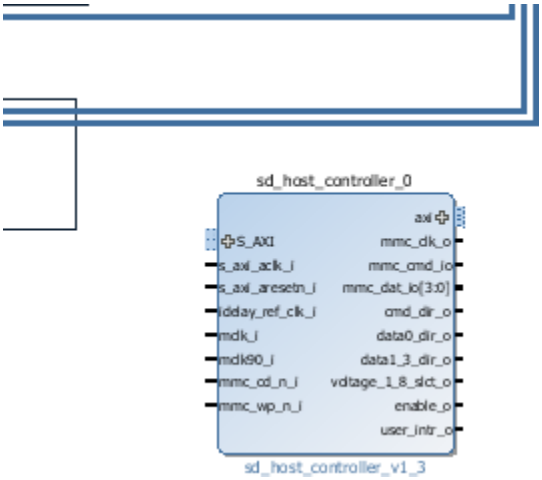

Figure 8: Step#6 to instantiate the iWave SD Host Controller IP in Design

7. Open the Zynq Processing system and enable the HP port, PL Interrupt and clock as shownbelow.

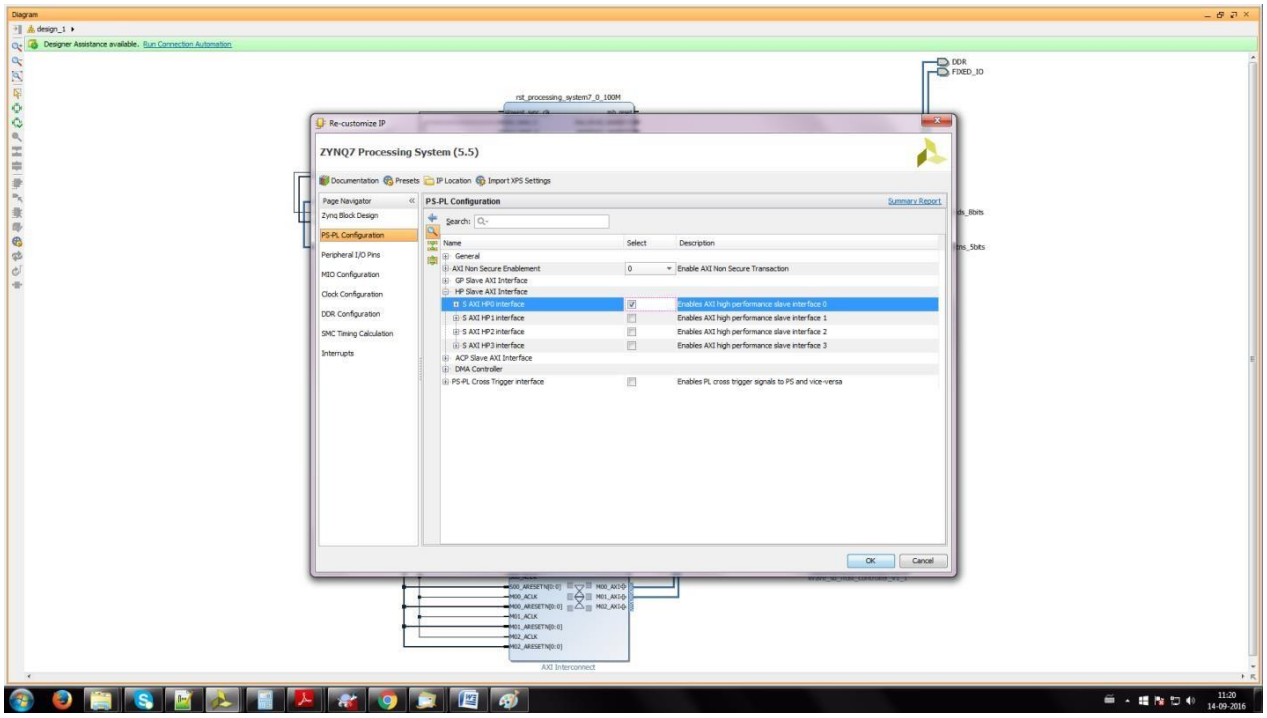

Figure 9: Step#7.1 to instantiate the iWave SD Host Controller IP in Design

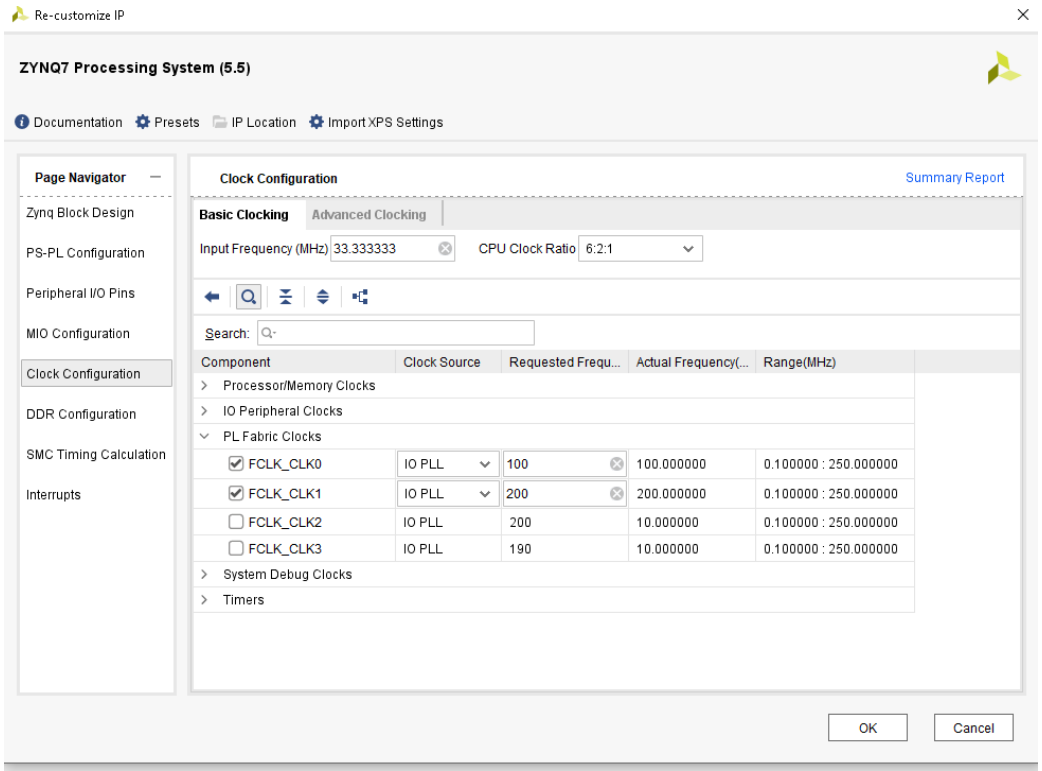

Figure 10: Step#7.2 to instantiate the iWave SD Host Controller IP in Design

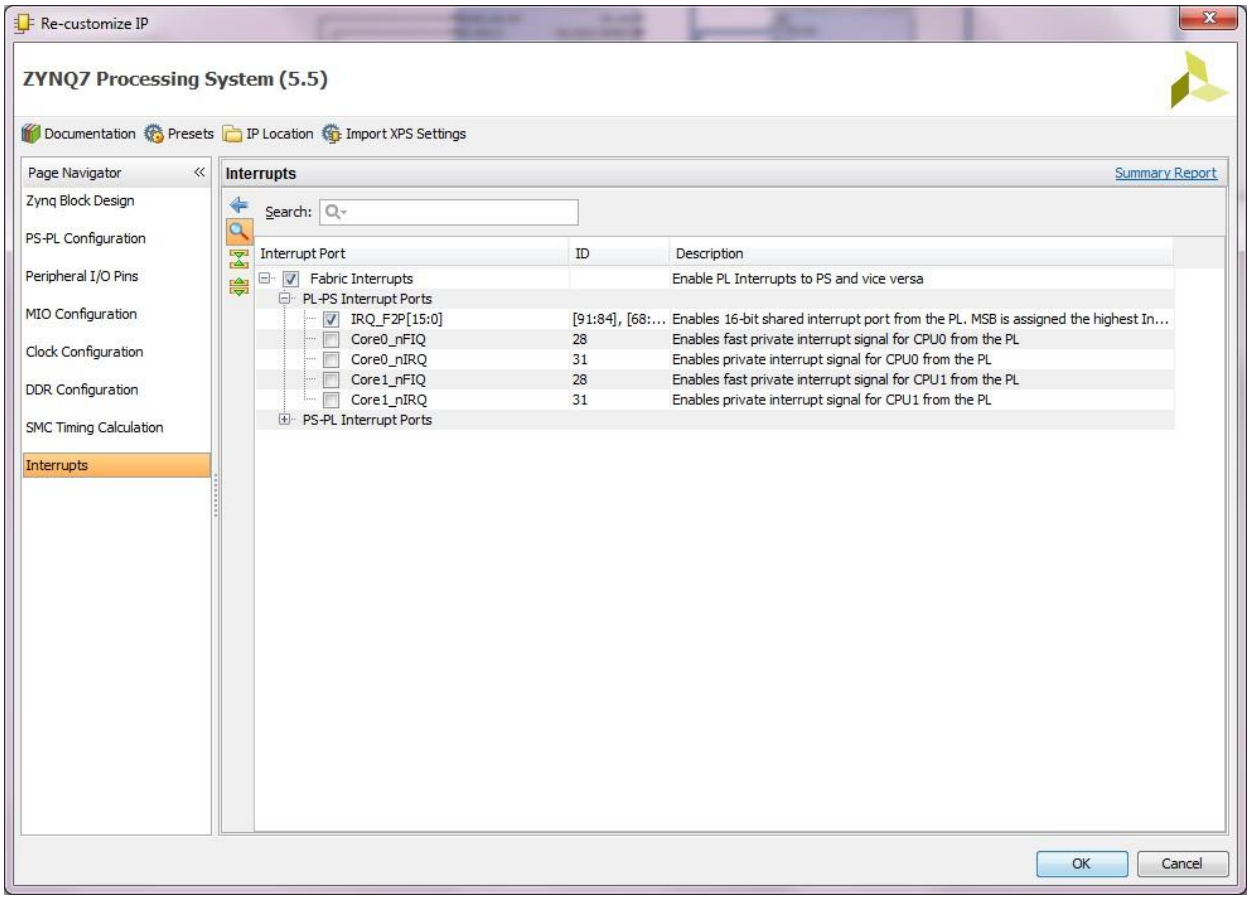

Figure 11: Step#7.3 to instantiate the iWave SD Host Controller IP in Design

8. Run the connection automation as shown below.

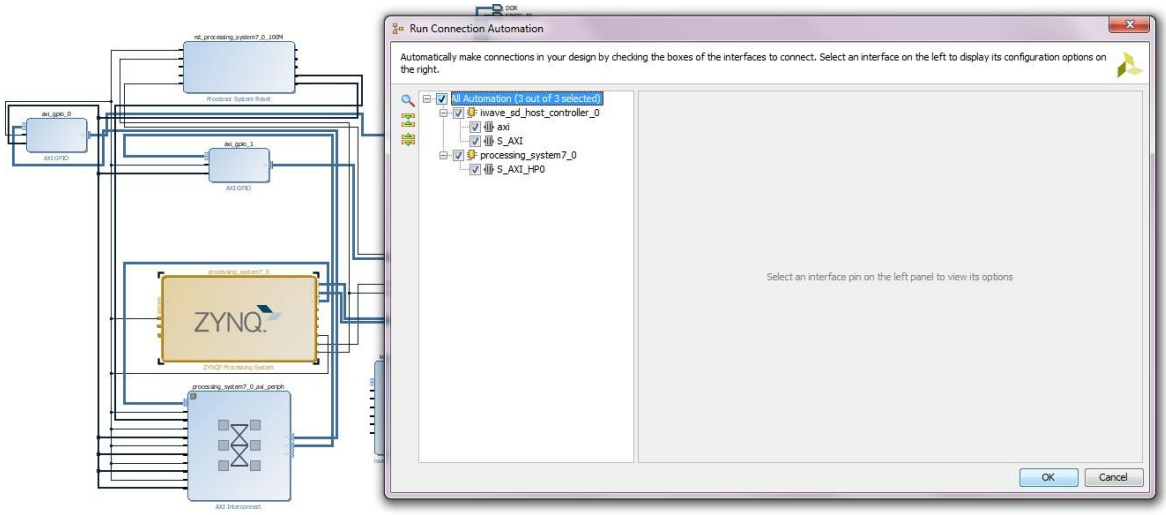

Figure 12: Step#8 to instantiate the iWave SD Host Controller IP in Design

9. Do the Clock and reset Connection as shown below[1]

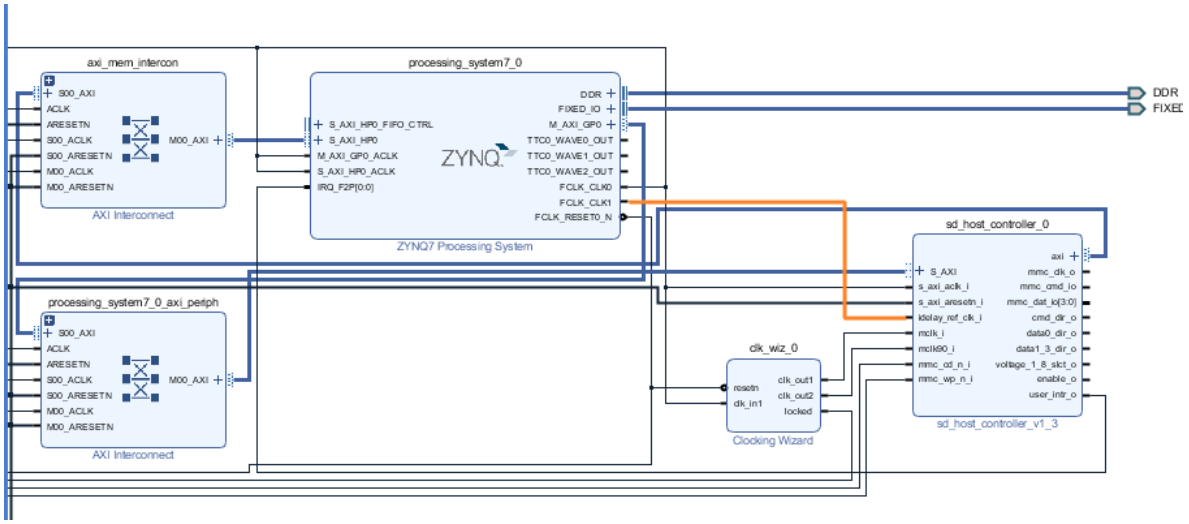

Figure 13: Step#9 to instantiate the iWave SD Host Controller IP in Design

10. Do Interrupt connection as shown below[2]

[1] mclk_i should be 208MHz(SD Base clock) and mclk90_i is 90 degree phase shifted mclk_i. Clock wizard is usedhere to generate the mclk_i and mclk90_i

[2] Interrupt is active high

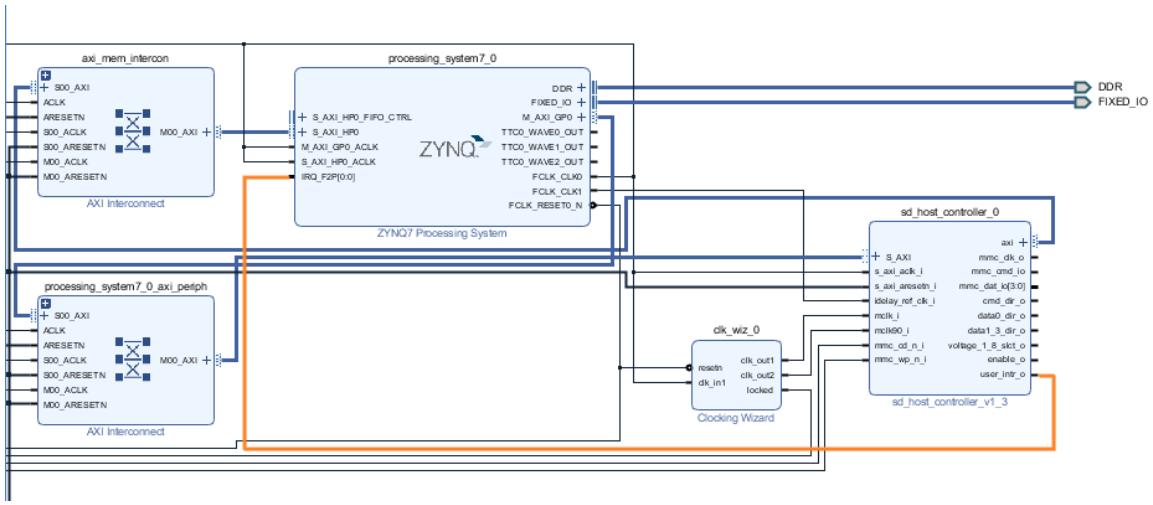

Figure 14: Step#10 to instantiate the iWave SD Host Controller IP in Design

11. Make the remaining port as external(Select the ports and press"Ctrl + T")

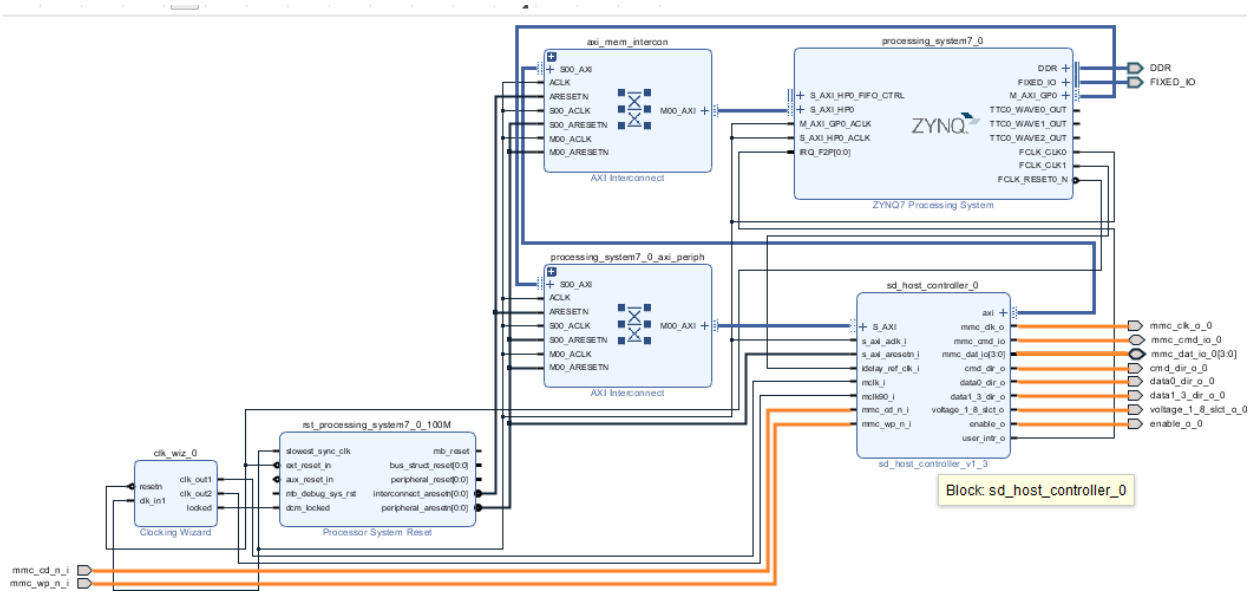

Figure 15: Step#11 to instantiate the iWave SD Host Controller IP in Design

12. Go to **Address Editor** Tab and make **Iwave_sd_host_controller_0** to range **4K** as shownbelow

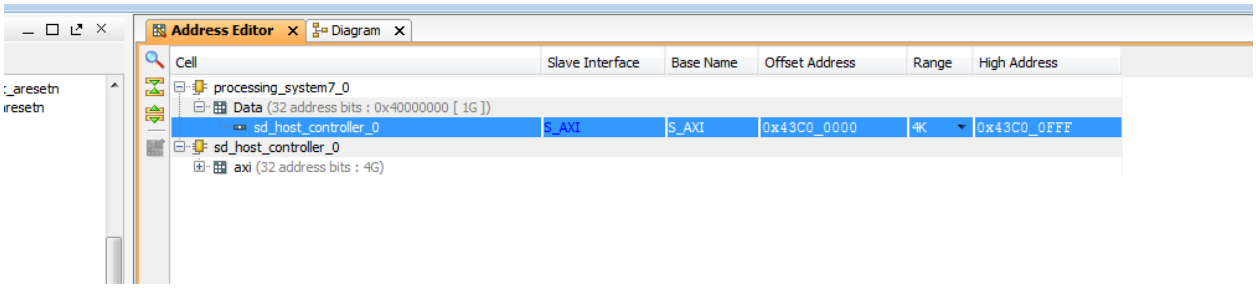

Figure 16: Step#12 to instantiate the iWave SD Host Controller IP in Design

13. Generate the output product as shown below

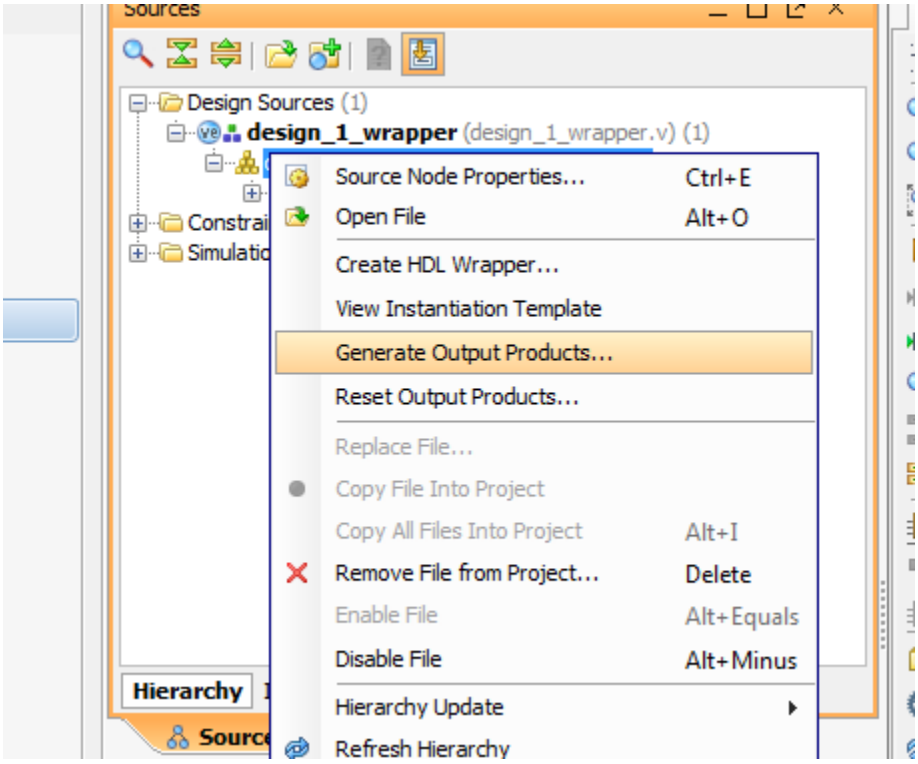

Figure 17: Step#13 to instantiate the iWave SD Host Controller IP in Design

14. Give the pin constraint for SD interface using xdc file
15. Generate the bit file.

3 Implementation Details

3.1 Clock Domain

SD Host Controller IP uses three clock domains SD Base clock (mclk_i and mclk90_i) and Hostprocessor interface Clock.

3.2 Signals Crossing Clock Domains

The following figure shows the constraints for the design. In the example design all three clocks is derived from PS. Tool automatically derives the period constraint for each clock. The constraints given here are IO standard constraint and False path constraints. These constraints arewritten in the .xdc file along with the clock and pin constraints.

```
#SD IO signal Pin constarints
set_property PACKAGE_PIN N22 [get_ports {mmc_dat_io[3]}]
set_property PACKAGE_PIN J21 [get_ports {mmc_dat_io[2]}]
set_property PACKAGE_PIN M21 [get_ports {mmc_dat_io[1]}]
set_property PACKAGE_PIN R19 [get_ports {mmc_dat_io[0]}]
set_property PACKAGE_PIN P17 [get_ports mmc_clk_o]
set_property PACKAGE_PIN T16 [get_ports mmc_cmd_io]
set_property PACKAGE_PIN N17 [get_ports data0_dir_o]
set_property PACKAGE_PIN P20 [get_ports data1_3_dir_o]
set_property PACKAGE_PIN J18 [get_ports cmd_dir_o]
set_property PACKAGE_PIN R20 [get_ports voltage_1_8_slct_o]
set_property PACKAGE_PIN L21 [get_ports mmc_cd_n_i]
set_property PACKAGE_PIN L22 [get_ports mmc_wp_n_i]
#SD IO Standard constraints
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 data0_dir_o]
set_property IOSTANDARD LVCMOS18 [get_ports data1_3_dir_o]
set_property IOSTANDARD LVCMOS18 [get_ports cmd_dir_o]
set_property IOSTANDARD LVCMOS18 [get_ports voltage_1_8_slct_o]
set_property IOSTANDARD LVCMOS18 [get_ports mmc_wp_n_i]
set_property IOSTANDARD LVCMOS18 [get_ports mmc_cd_n_i]
```

Figure 18: Pin and IO standard constraints

```
set_false_path -from [get_clocks clk_fpga_0] -to [get_clocks clk_fpga_1]
set_false_path -from [get_clocks clk_fpga_0] -to [get_clocks -of_objects [get_pins design_1_wrapper/design_1_i/clk_wiz_0/inst/plle2_adv_inst/CLKOUT1]]
set_false_path -from [get_clocks -of_objects [get_pins design_1_wrapper/design_1_i/clk_wiz_0/inst/plle2_adv_inst/CLKOUT1]] -to
[get_clocks -of_objects [get_pins design_1_wrapper/design_1_i/clk_wiz_0/inst/plle2_adv_inst/CLKOUT0]]
set_false_path -from [get_clocks -of_objects [get_pins design_1_wrapper/design_1_i/clk_wiz_0/inst/plle2_adv_inst/CLKOUT1]] -to
[get_clocks clk_fpga_0]
set_false_path -from [get_pins design_1_wrapper/design_1_i/sd_host_controller_0/inst/mmc_host_inst_9/rstall_n_r2_reg/C]
```

Figure 19: False path constraints

4 FPGA Implementation

4.1 Resource Utilization

The table below shows the resource utilization summary for Xilinx Zynq 7000 xc7z020clg484-1device with SD 3.0 Host controller IP.

Table 2: Resource Utilization

LUTs	2671/53200	5.02%
FFs	2202/106400	2.07%
Slices	1804/13300	8.15%