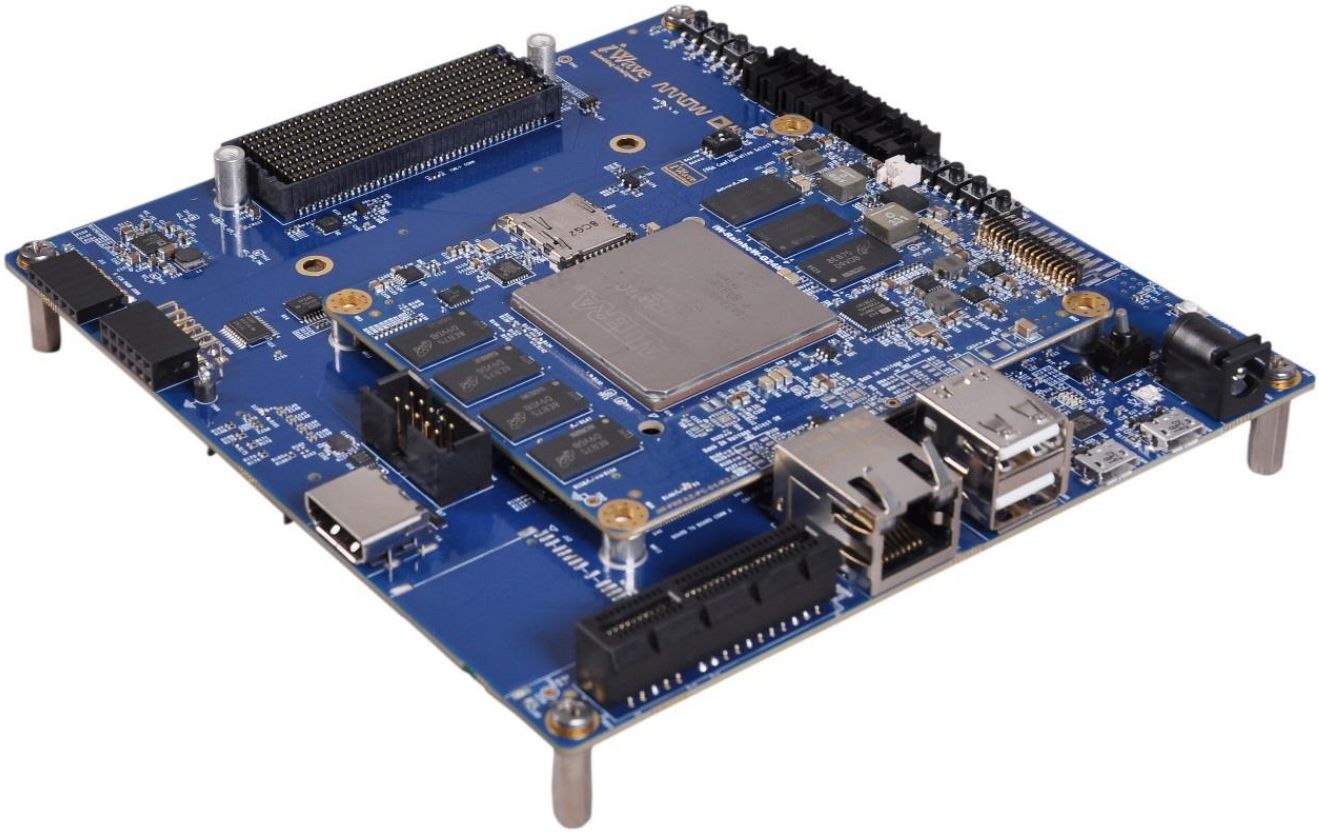
Arria10 SoC/FPGA FMC+ Development platform

FPGA User Guide



iWave

Embedding Intelligence

Document Revision History

Document Number		iW-PRFXR-Arria10_SoC_FPGA_DevKit-FPGAUserGuide-REL0.1
Revision	**Date**	**Description**
0.1	15th May 2019	Draft Version

Disclaimer

iWave Systems reserves the right to change details in this publication including but not limited to any Product specification without notice.

No warranty of accuracy is given concerning the contents of the information contained in this publication. To the extent permitted by law no liability (including liability to any person by reason of negligence) will be accepted by iWave Systems, its subsidiaries or employees for any direct or indirect loss or damage caused by omissions from or inaccuracies in this document.

CPU and other major components used in this product may have several silicon errata associated with it. Under no circumstances, iWave Systems shall be liable for the silicon errata and associated issues.

Trademarks

All registered trademarks and product names mentioned in this publication are used for identification purposes only.

Certification

iWave Systems Technologies Pvt. Ltd. is an ISO 9001:2015 Certified Company.

Warranty & RMA

Warranty support for Hardware: 1 Year from iWave or iWave's EMS partner.

For warranty terms, go through the below web link,

http://www.iwavesystems.com/support/warranty.html

For Return Merchandise Authorization (RMA), go through the below web link,

http://www.iwavesystems.com/support/rma.html

Technical Support

iWave Systems technical support team is committed to provide the best possible support for our customers so that our Hardware and Software can be easily migrated and used.

For assistance, contact our Technical Support team at,

Email : support.ip@iwavesystems.com

Website : www.iwavesystems.com

Address : iWave Systems Technologies Pvt. Ltd.
7/B, 29th Main, BTM Layout 2nd Stage,
Bangalore, Karnataka,
India – 560076

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

1.1 Purpose

This document is intended as a guide to assist anyone who is making use of the board to understand the overall process of compiling the hardware design and programming the iW-RainboW-G24M-Arria10 FMC+ development platform and to understand the high-level FPGA implementation.

1.2 Overview

Arria® 10 SoC delivers optimal performance, power efficiency, small form factor, and low cost for midrange applications. The Arria 10 SoC, combines a dual-core ARM* Cortex*-A9 MPCore* Hard Processor System (HPS) with industry-leading programmable logic technology. The Arria 10 SoC offers a processor with a rich feature set of embedded peripherals, embedded high-speed transceivers, hard memory controllers, and protocol intellectual property (IP) controllers - all in a single highly integrated package.

1.3 List of Acronyms

The following acronyms will be used throughout this document.

Table 1: Acronyms & Abbreviations

Acronyms	Abbreviations
FPGA	Field Programmable Gate Array
GND	Ground
FMC	FPGA Mezzanine Card
PLL	Phase Locked Loop
Soc	System On Chip
JTAG	Joint test action group
SOM	System On Modules
HPS	Hard Processor System
MGT	Multi Gigabit Transceiver
DDR	Double Data Rate
PCIe	Peripheral Component Interconnect Express
s/w	Switch
SFP	Small factor pluggable
QSPI	Quad Serial Peripheral Interface

2. FPGA Design

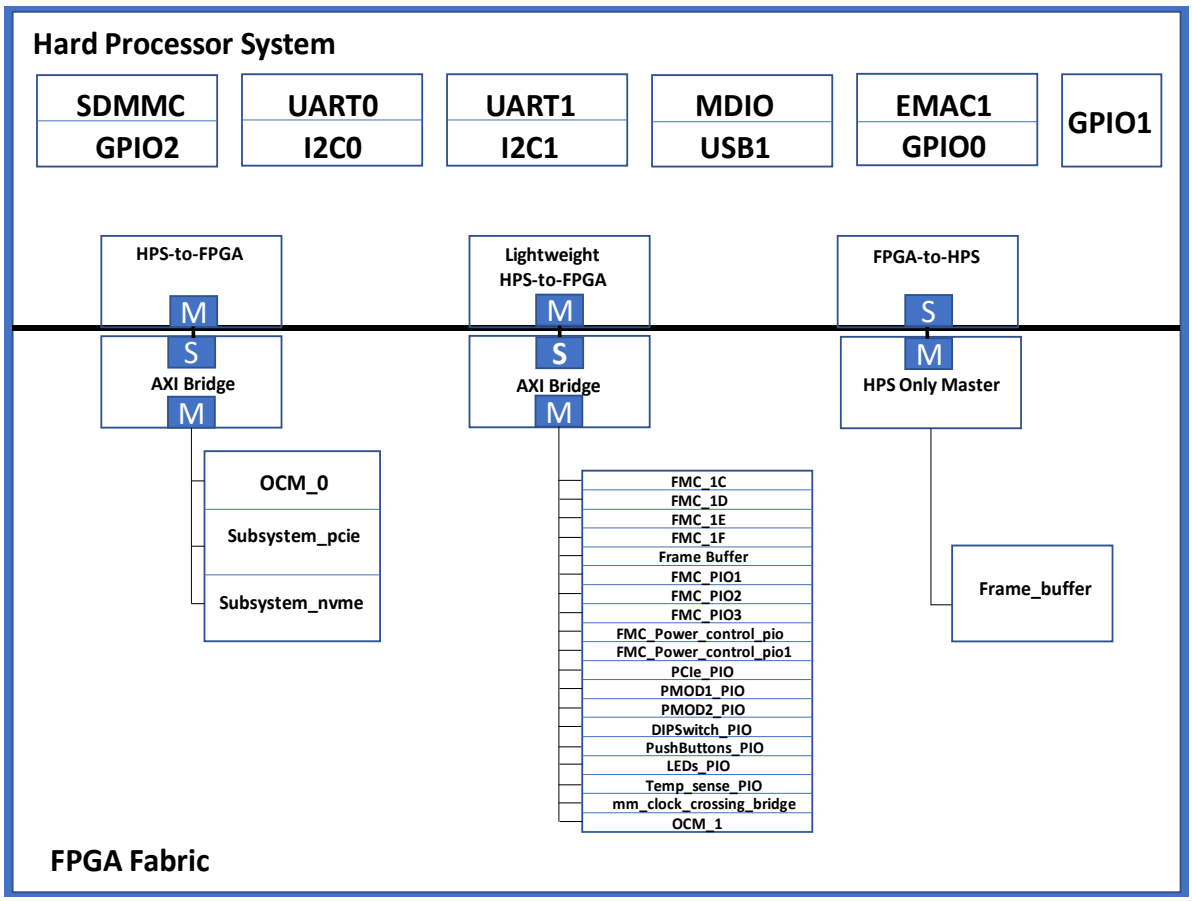


Figure 1: FPGA Design Block Diagram

2.1 FPGA Design Overview

2.1.1 Hard Processor System

The Intel® Arria10 system-on-a-chip (SoC) is composed of two distinct portions - a dual-core ARM Cortex-A9 hard processor system (HPS) and an FPGA. The HPS architecture integrates a wide set of peripherals that reduce board size and increase performance within a system. A dedicated security manager within the HPS supports secure boot with the ability to authenticate, decrypt and provide tamper event response.

The SoC features the following types of I/O Pins:

- Dedicated I/O - I/O that is dedicated to an external non-volatile storage device (for boot ROM), HPS clock and resets.
- Shared I/O – I/O that can be assigned to peripherals in the HPS or FPGA logic.
- FPGA I/O – I/O that is dedicated to the FPGA Fabric.

2.1.2 HPS-to-FPGA Bridge

The memory map of soft IP peripherals, as viewed by the microprocessor unit (MPU) of the Hard Processor System, starts at HPS-to-FPGA base address of 0xC000_0000.

Components connected to HPS-to-FPGA AXI Bridge:

OCM_0:

- 256 KB of On-Chip-Memory provided for PCIe environment.

Subsytem_pcie:

PCIe subsystem environment connected to the ARM for controlling the PCIe interface. Subsystem_pcie consists of the below components.

- Arria10 Hard IP for PCI Express
- Avalon-mm Pipeline Bridge IP
- Altera MSI to GIC generator IP
- Modular Scatter-Gather DMA IP
- Address Span Extender IP
- Avalon-mm Clock Crossing Bridge IP

Subsytem_nvme:

NVMe subsystem environment connected to the ARM for controlling the NVMe interface. Subsystem_nvme consists of the below components.

- Arria10 Hard IP for PCI Express
- Avalon-mm Pipeline Bridge IP
- Altera MSI to GIC generator IP
- Modular Scatter-Gather DMA IP
- Address Span Extender IP
- Avalon-mm Clock Crossing Bridge IP

2.1.3 Lightweight HPS-to-FPGA Address Map

The memory map of system peripherals in the FPGA portion of the SoC as viewed by the MPU (Cortex-A9), which starts at the lightweight HPS-to-FPGA base address of **0xFF20_0000**.

Components connected to Lightweight HPS-to-FPGA AXI Bridge:

FMC_1C:

Transceiver subsystem connected to the ARM for controlling Bank1C 4 transceivers channels transceivers. These transceivers can be controlled by ARM as well as by FPGA using transceiver toolkit. 100MHz reference clock is used and Line rate upto 12 Gbps is supported. The transceiver subsystem consists of the below components.

- Avalon-mm Data Generator IP
- Avalon-mm Data Checker IP
- Transceiver Native PHY IP
- Transceiver FPLL IP
- Transceiver PHY Reset Controller IP

FMC_1D:

Transceiver subsystem connected to the ARM for controlling Bank 1D 4 transceivers channels transceivers. These transceivers can be controlled by ARM as well as by FPGA using transceiver toolkit. 100MHz reference clock is used and Line rate upto 12 Gbps is supported. The transceiver subsystem consists of the below components.

- Avalon-mm Data Generator IP
- Avalon-mm Data Checker IP
- Transceiver Native PHY IP
- Transceiver FPLL IP
- Transceiver PHY Reset Controller IP

FMC_1E:

Transceiver subsystem connected to the ARM for controlling Bank 1E 4 transceivers channels transceivers. These transceivers can be controlled by ARM as well as by FPGA using transceiver toolkit. 100MHz reference clock is used and Line rate upto 12 Gbps is supported. The transceiver subsystem consists of the below components.

- Avalon-mm Data Generator IP
- Avalon-mm Data Checker IP
- Transceiver Native PHY IP
- Transceiver FPLL IP
- Transceiver PHY Reset Controller IP

FMC_1F:

Transceiver subsystem connected to the ARM for controlling Bank 1F 4 transceivers channels transceivers. These transceivers can be controlled by ARM as well as by FPGA using transceiver toolkit. 100MHz reference clock is used and Line rate upto 12 Gbps is supported. The transceiver subsystem consists of the below components.

- Avalon-mm Data Generator IP
- Avalon-mm Data Checker IP
- Transceiver Native PHY IP
- Transceiver FPLL IP
- Transceiver PHY Reset Controller IP

NOTE: To test the design only by FPGA using the Transceiver Toolkit, refer section Appendix.

Frame_Buffer:

Frame buffer module is connected to the ARM for providing progressive video output. The data is fed from the HPS DDR to the frame buffer which is connected to a Clocked Video Output module to provide a progressive video output to an HDMI port.

FMC_PIO1:

- 32-bits Bi-Directional Parallel I/O are connected to the ARM for controlling FMC+ GPIO's.

FMC_PIO2:

- 32-bits Bi-Directional Parallel I/O are connected to the ARM for controlling FMC+ GPIO's.

FMC_PIO3:

- 31-bits Bi-Directional Parallel I/O are connected to the ARM for controlling FMC+ GPIO's.

FMC_Power_control_pio:

- 4-bits Output Parallel I/O are connected to the ARM for controlling FMC+ power sequence.

FMC_Power_control_pio1:

- 2-bits Input Parallel I/O are connected to the ARM for controlling FMC+ power sequence.

PCIe_PIO:

- 4-bits Bi-Directional Parallel I/O are connected to the ARM for controlling PCIe GPIO's.

PMOD1_PIO:

- 8-bits Bi-Directional Parallel I/O are connected to the ARM for controlling PMOD GPIO's.

PMOD2_PIO:

- 4-bits Bi-Directional Parallel I/O are connected to the ARM for controlling PMOD GPIO's.

DIPSwitch_PIO:

- 4-bits Input Parallel I/O are connected to the ARM for controlling dip switches logic.

PushButtons_PIO:

- 4-bits Input Parallel I/O are connected to the ARM for controlling push buttons logic.

LEDs_PIO:

- 4-bits Output Parallel I/O are connected to the ARM for controlling LED glowing.

Temp_sense_PIO:

- 10-bits Input Parallel I/O are connected to the ARM. The output value from the Temperature sensor module is passed to the software for reading.

mm_clock_crossing_bridge:

- For transferring the data between different clock domains Avalon clock crossing bridge is used where it transfers the data between HPS to subsystem_pcie and subsystem_nvme.

OCM_1:

- 256 KB of On-Chip-Memory provided for NVMe environment.

2.2 FPGA Components Address Map

Address Mapping of the components can be found in the below attached figure.

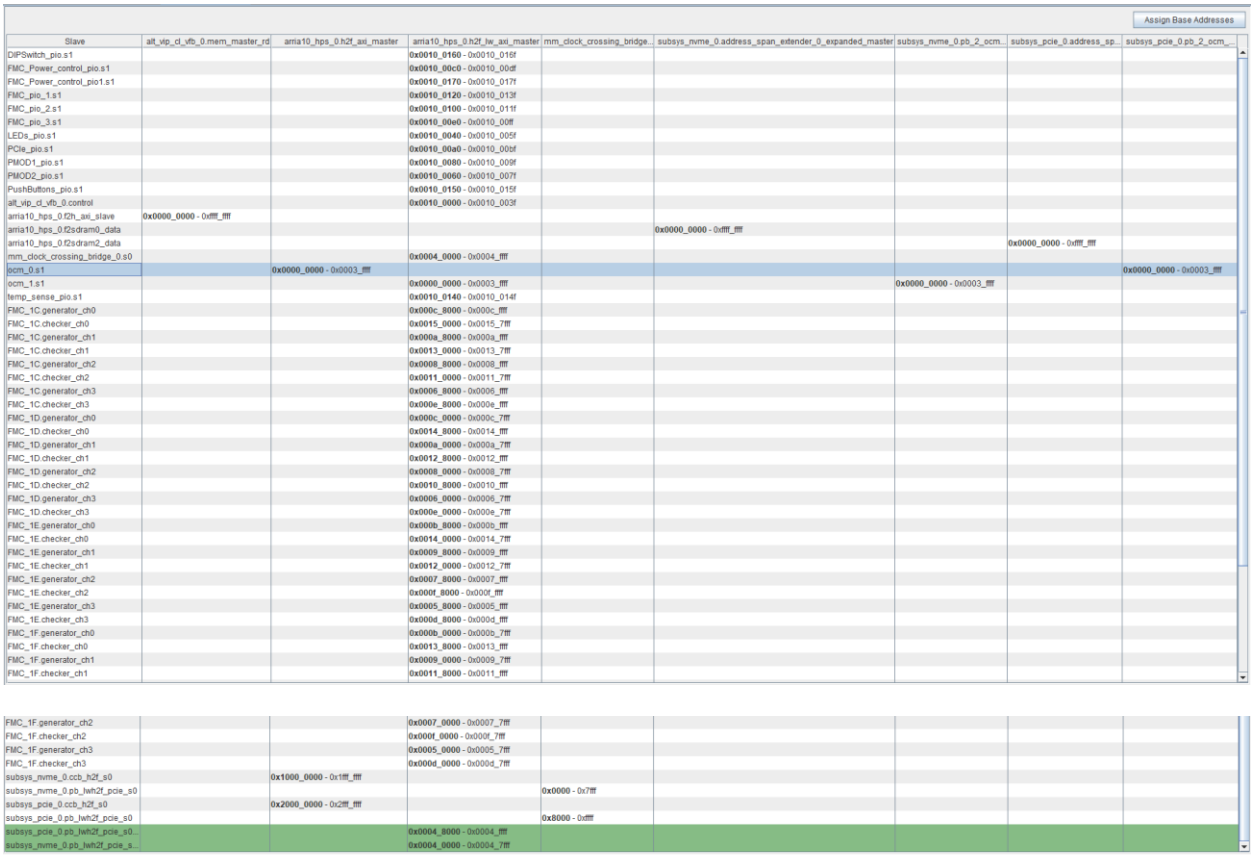


Figure 2: Address Mapping

3. Hardware Design Compilation

3.1 Overview

The following steps provides the procedure to re-compile the hardware design. This may be needed in case any changes are made to the design. Otherwise the pre-compiled design can be used as-is.

3.2 Prerequisites

- Altera Quartus 18.1 Prime Pro version software needs to be installed in the Host PC.
- The Altera USB Blaster II cable
- Alternatively, we can directly connect the micro-cable to the On-Board USB Blaster to program the FPGA.

3.3 Hardware Design Compilation

- Retrieve the archived Example design from the Release provided and save it to your local disk.
- Extract the example design (iW-PRFXR-SY-01-RX.X-RELX.X-SX480.tar.gz) provided to your local folder.
- Start Quartus 18.1 Prime Pro Tool.

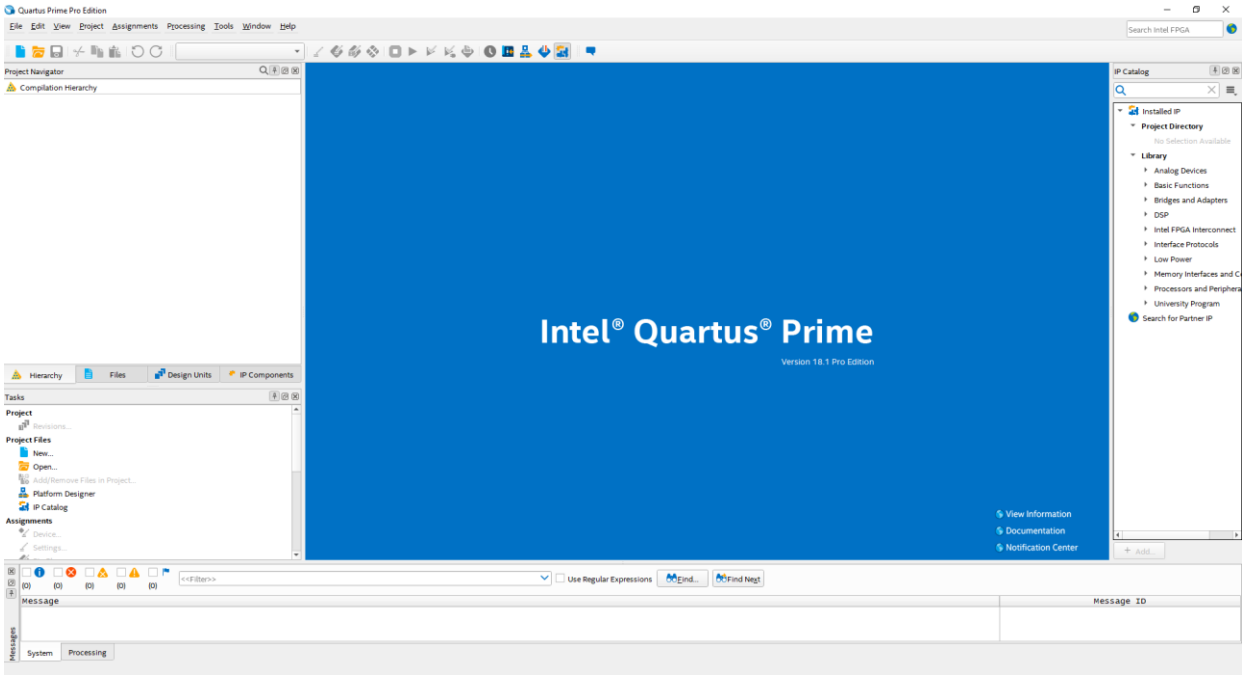


Figure 3: Quartus 18.1 Prime Pro Tool

- In Quartus, open the hardware project by going to File → Open Project. Browse to the local folder where the example design is extracted and select the ".qpf" file and click Open.

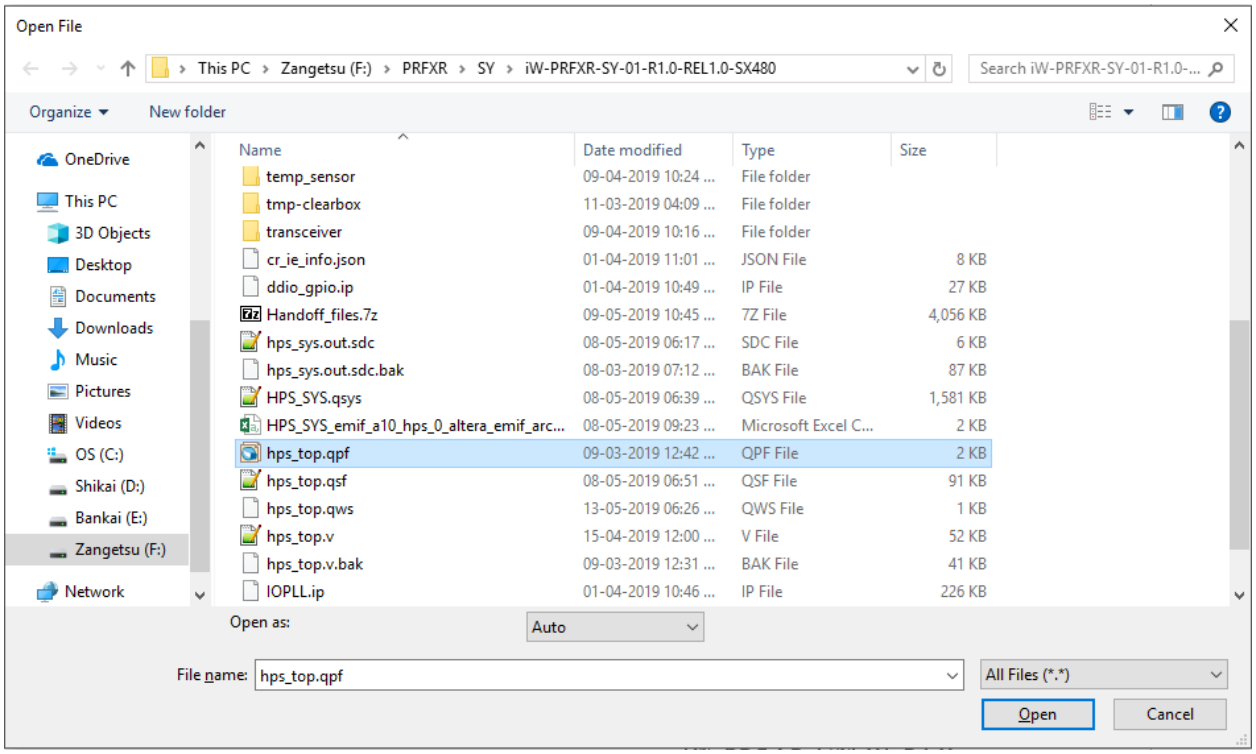


Figure 4: Loading .qpf file

- In Quartus, start Platform Designer by going to Tools → Platform Designer.
- The tool will ask to select a file to open. Select the file ~/iW-PRFXR-SY-01-RX.X-RELX.X-SX480/HPS_SYS.qsys and click Open.

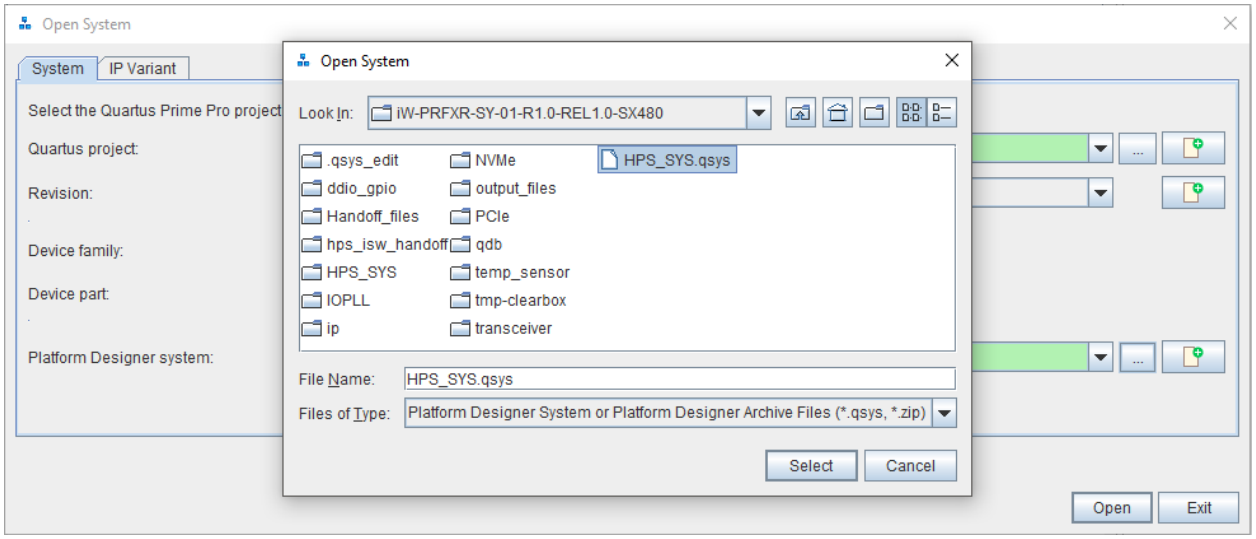


Figure 5: Loading Platform designer file

- Platform Designer will load the file, displaying a progress bar. Once file is loaded, Platform Designer will display the system.

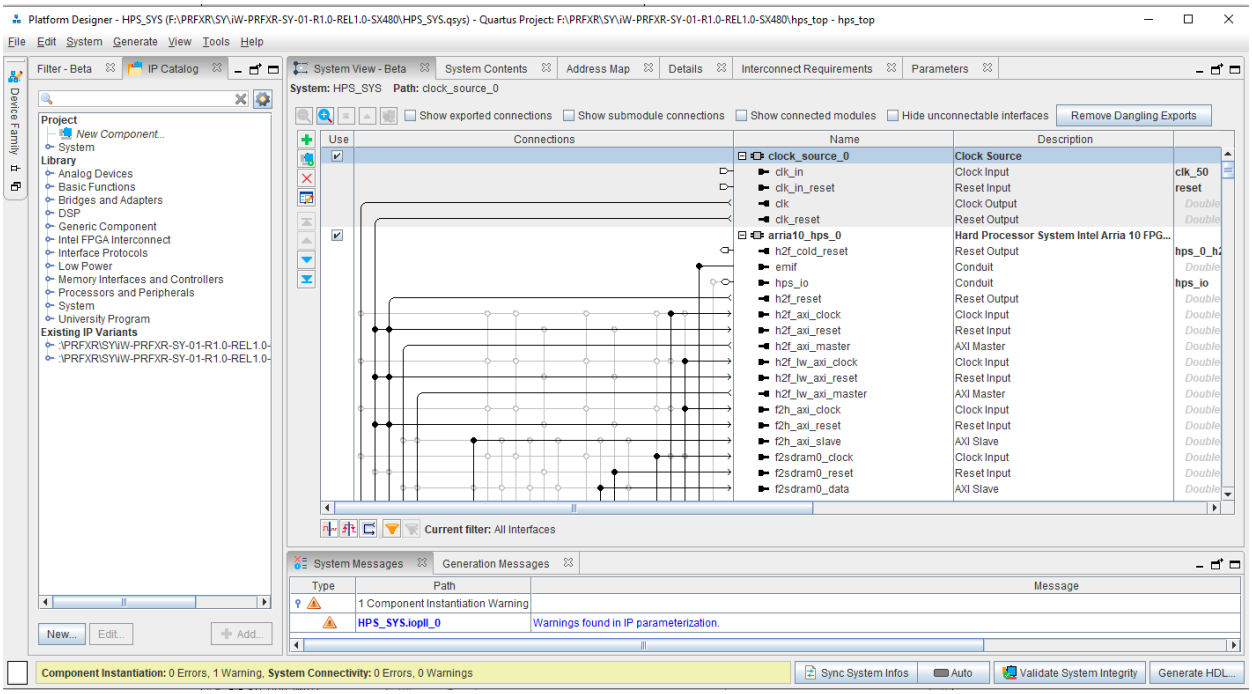


Figure 6: Platform Designer

- If any changes are done in the design then, in Platform Designer, click the "Sync All System Infos", then "Validate System Integrity" and "Reload and Update All Components" to refresh the IP. Then click the Generate HDL ... button on the bottom right corner. The Generation window will open. Click Generate button on the bottom right corner.

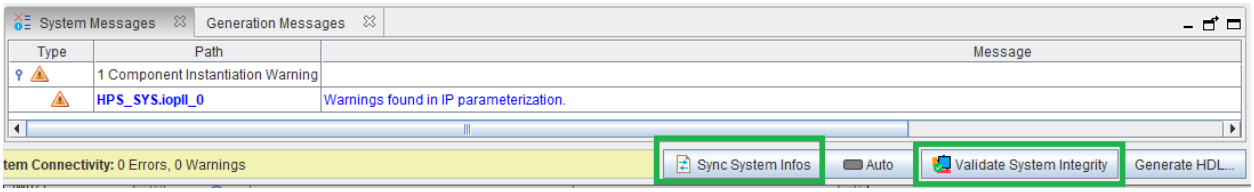


Figure 7: Validating System Integrity

- Platform Designer will generate the system, displaying a progress bar.

- Once complete, Platform Designer will display the "Generate Completed" message. Click Close to close the Generate window and get back to main Platform Designer window.

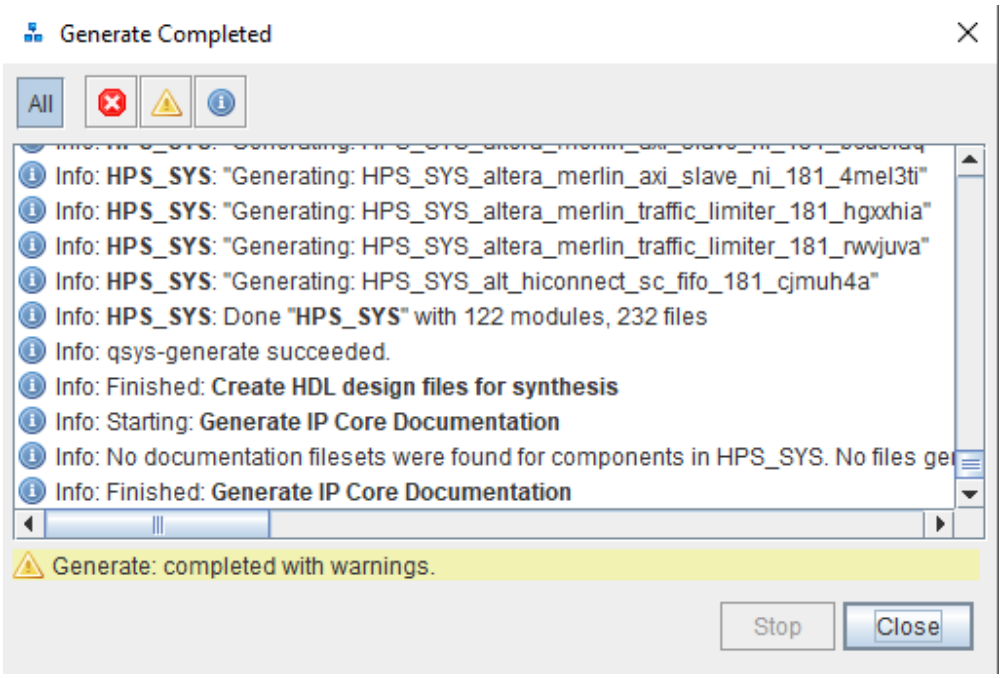


Figure 8: Qsys Generation Completion

- Click Finish to close the Platform Designer window and get back to main Quartus window.
- In Quartus, start a compilation by going to Processing → Start Compilation. Then Quartus will compile the project.
- When compilation is completed, Quartus will display the status.

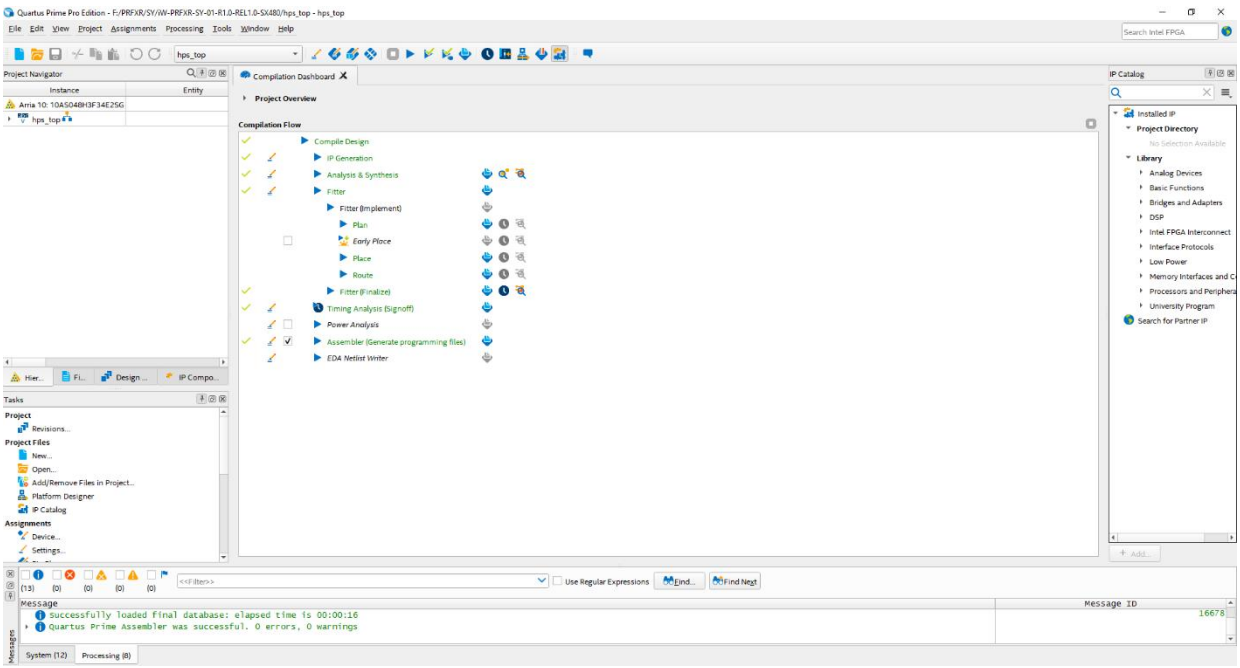


Figure 9: Recompilation Completed

3.4 Programming File Creation (.sof to .rbf format).

- In your Main page of the tool, Go to Files → Convert Programming Files.

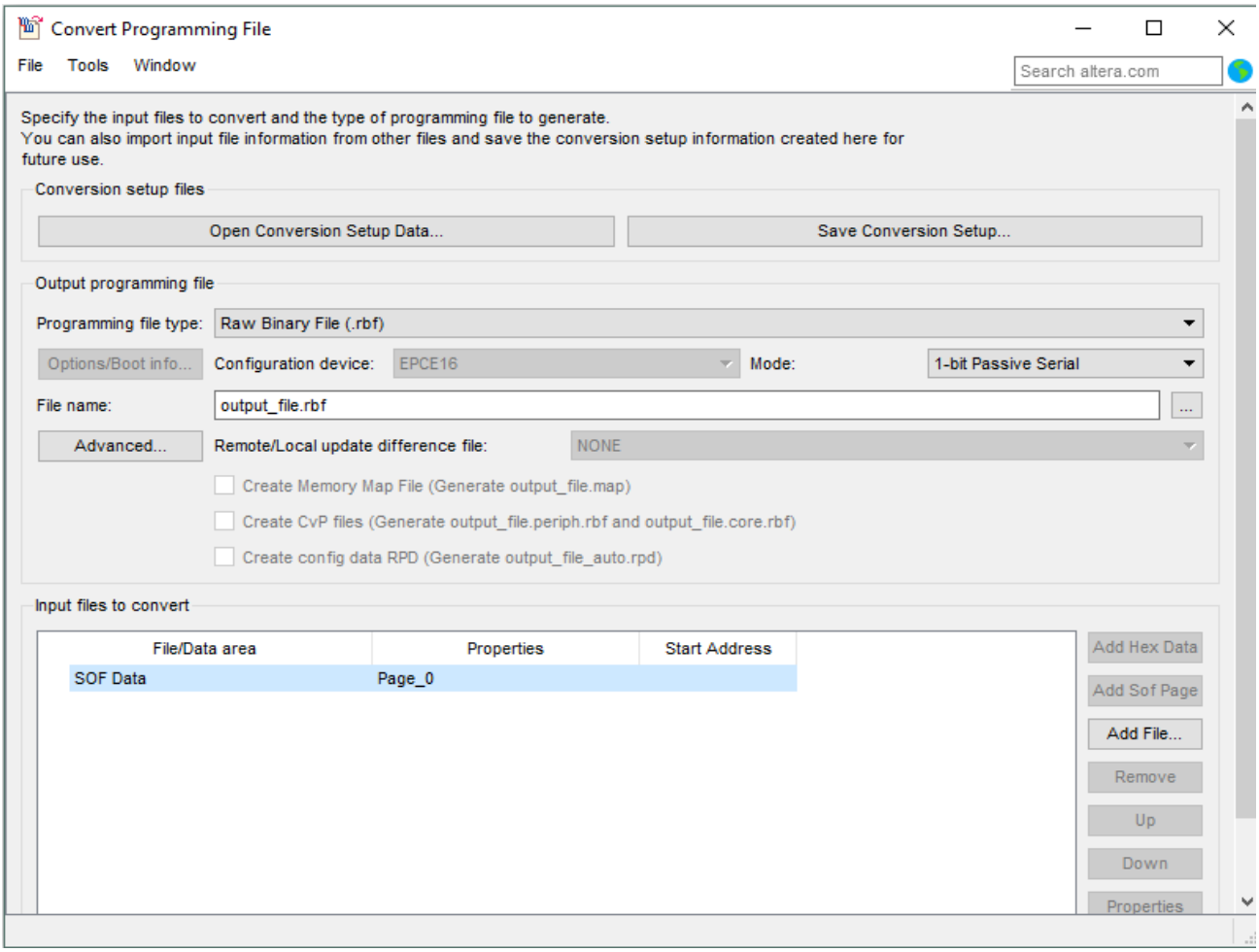


Figure 10: Convert Programming File window

- In the Convert Programming Files window, Under Programming file type select Raw Binary Files (.rbf) from the dropdown list.

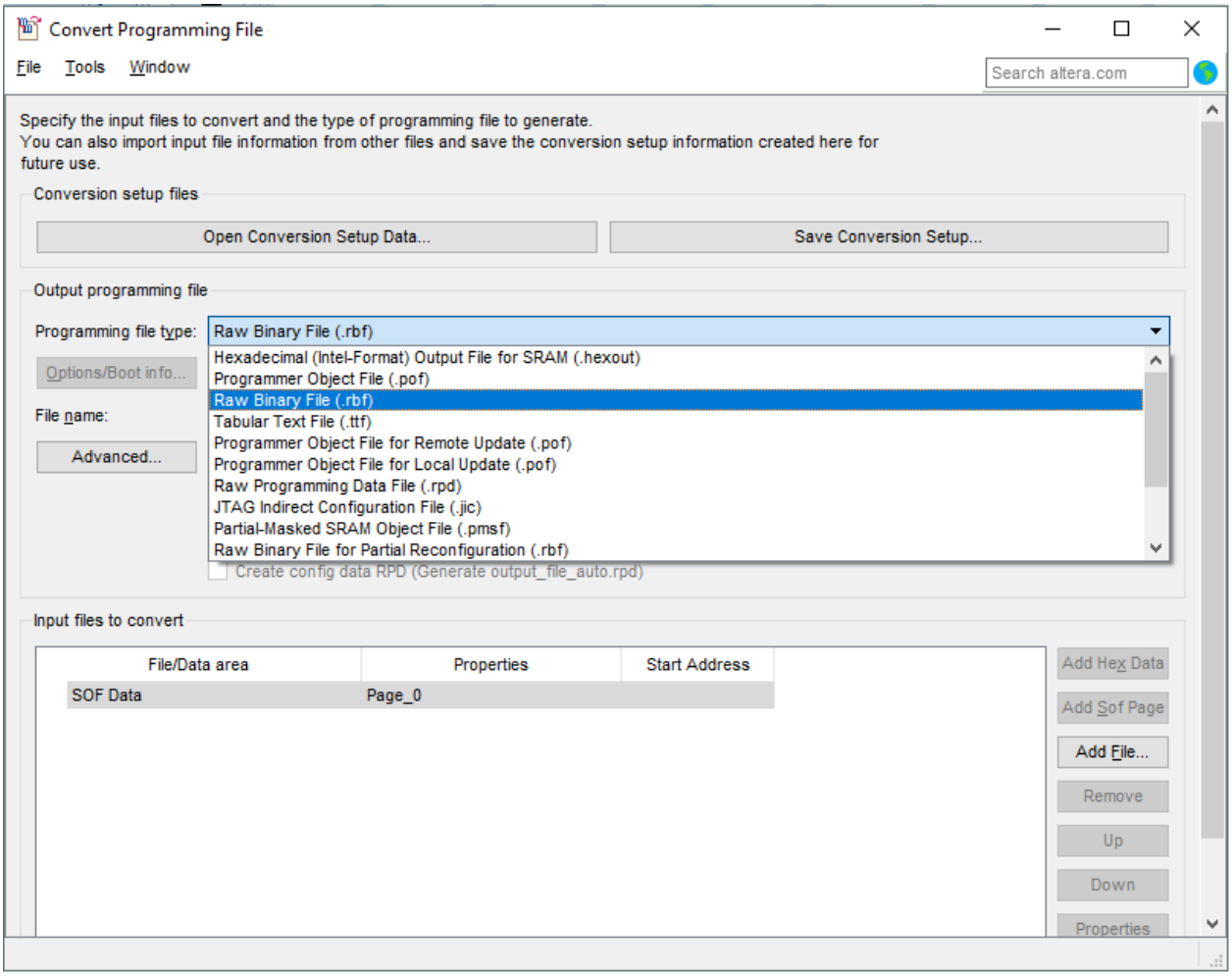


Figure 11: Selecting Programming file type

- Provide the name of your output file.
- In the Input files to convert area add your .sof file which is to be converted.

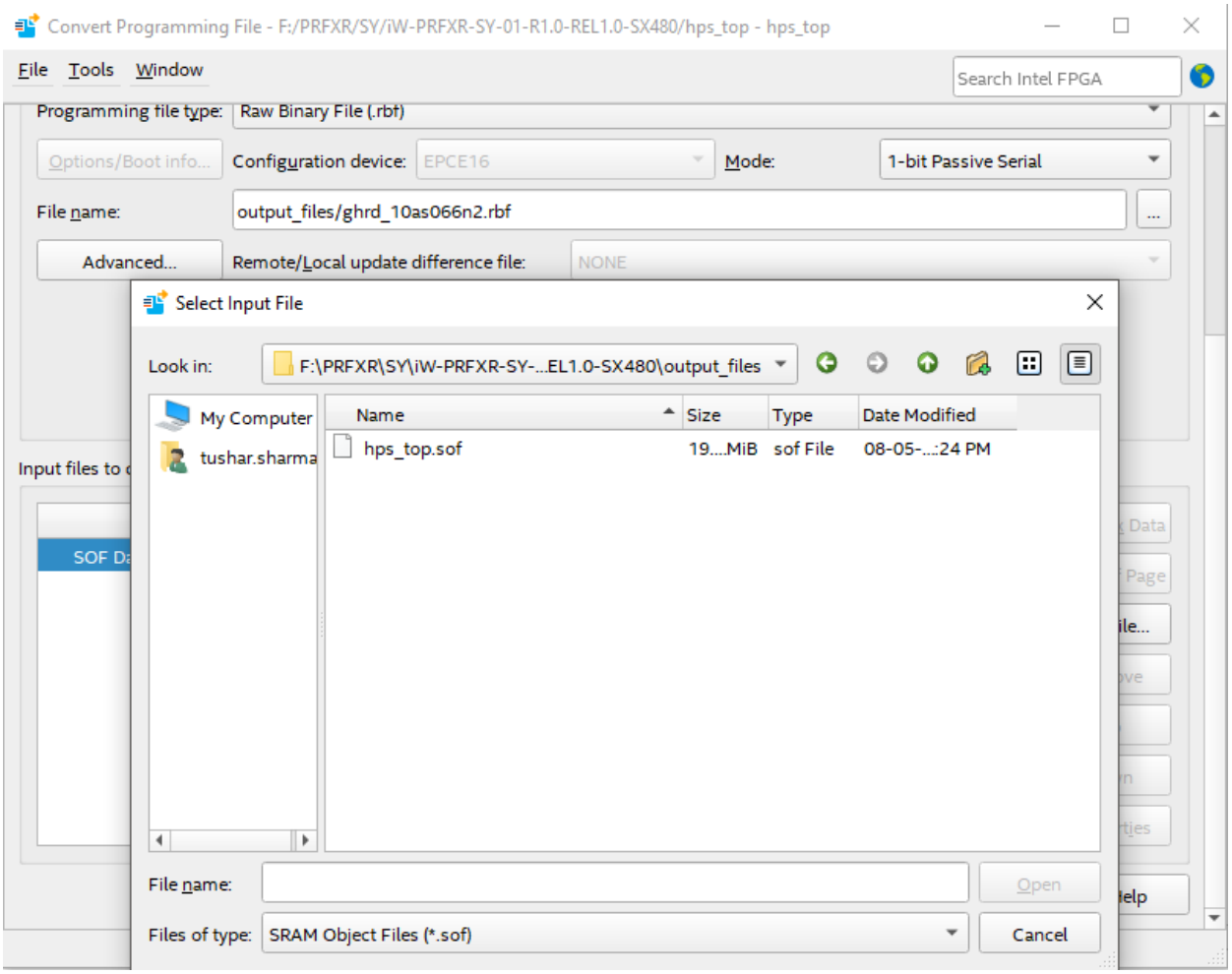


Figure 12: Adding .sof file for the conversion

- After adding your .sof file, click Generate to generate your .rbf file.

3.5 FPGA Programming

To program the FPGA independently instead of loading from Micro SD, follow the below procedure.

- Switch on the board.
- Open the Altera Quartus18.1 Prime Pro software. The following window will get open.

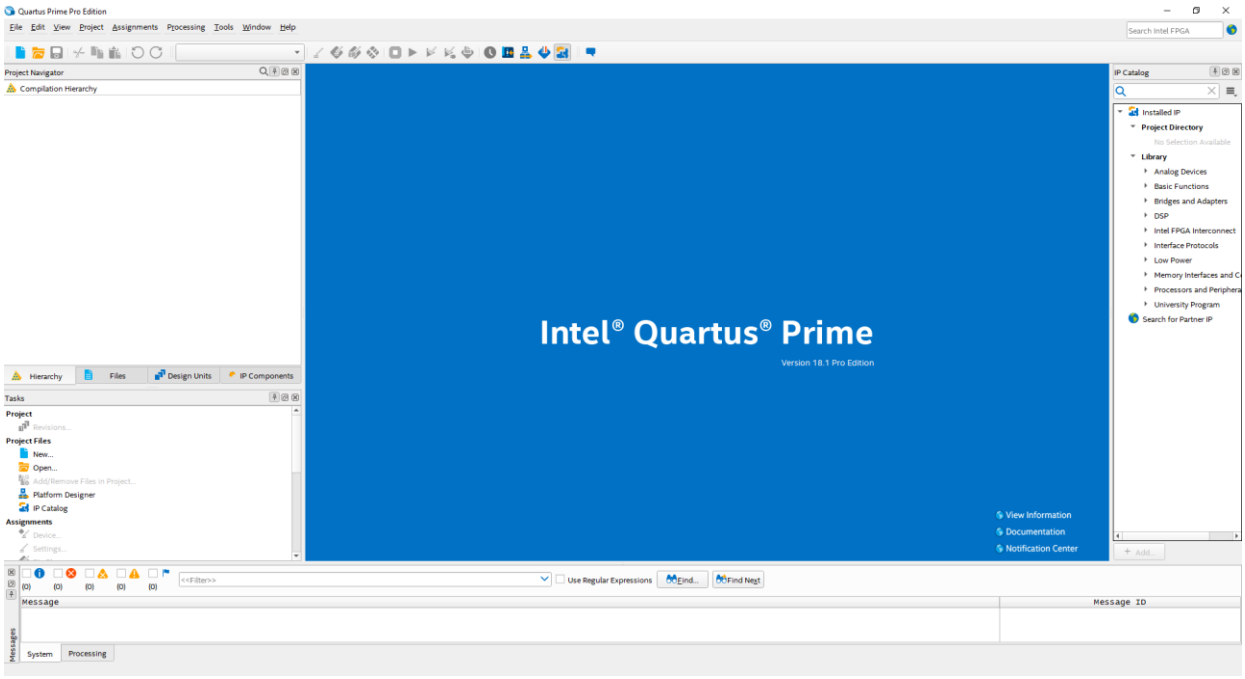


Figure 13: Startup page of Quartus Prime Software

- Select File → Open → iW-PRFXR-SY-01-RX.X-RELX.X-SX480

Note: Here FPGA design for FMC interfaces test is taken as the example design to program

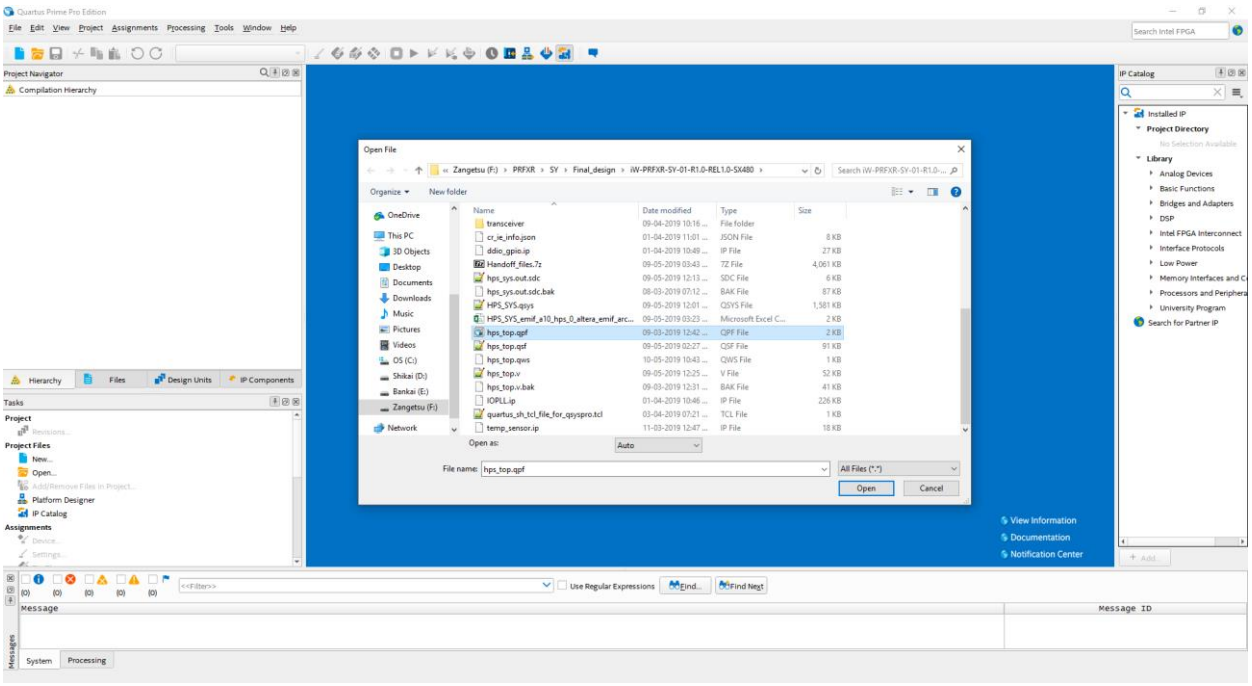


Figure 14: Project loading - 1

- Select the hps_top.qpf file to open the example design.

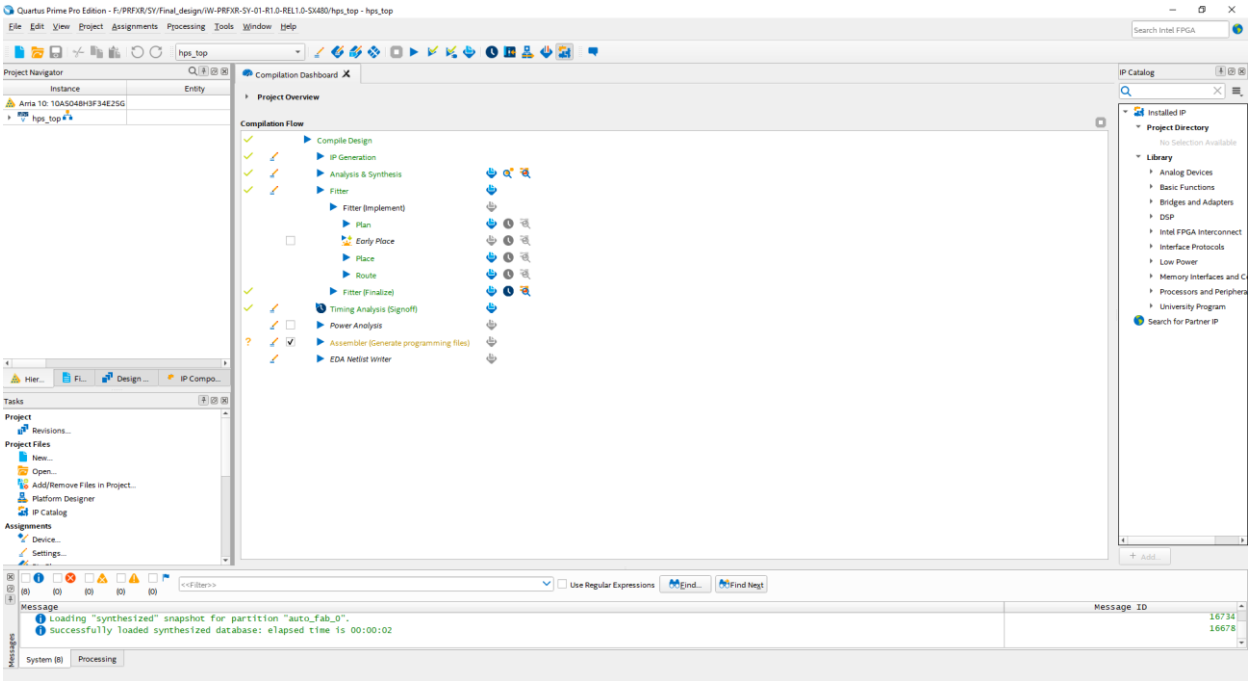


Figure 15: Project loading - 2

- Go to Tools → Programmer.

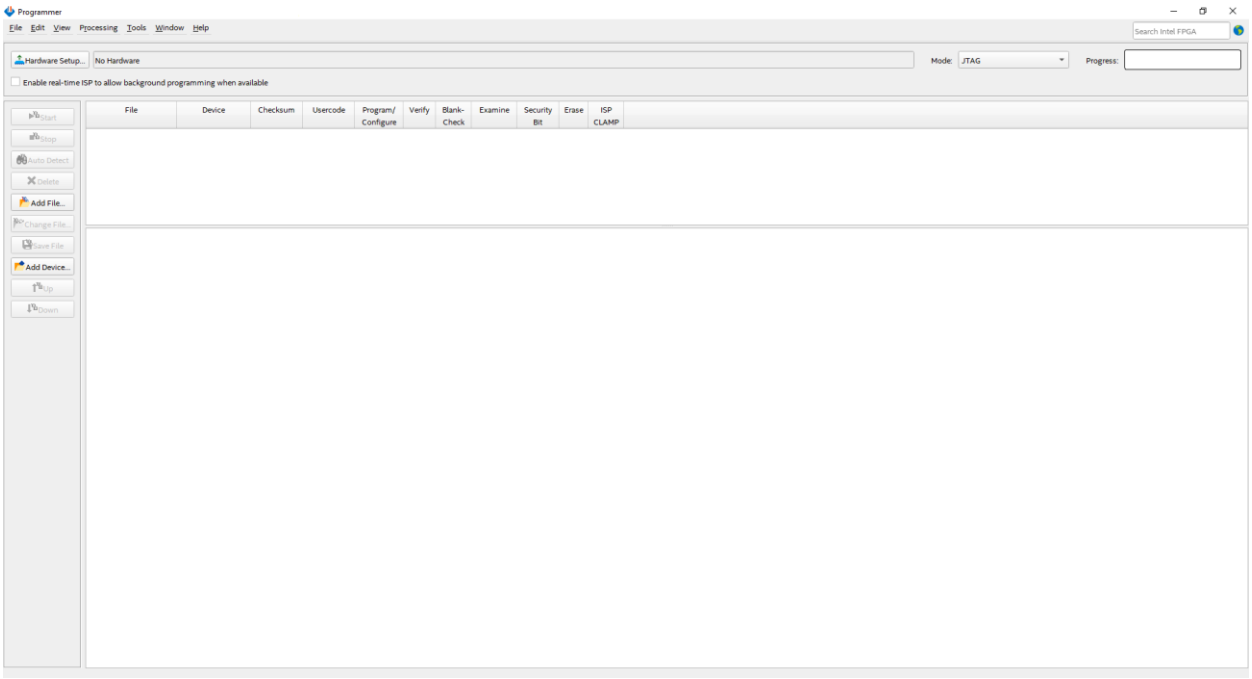


Figure 16: Programming Project - 3

- On selecting the Programmer, the above page is displayed.
- By default, JTAG will be detected. If not detected, try removing the JTAG cable from your system, re-insert the cable.

- On the Programmer page select the Hardware Setup option, then select the JTAG connection.

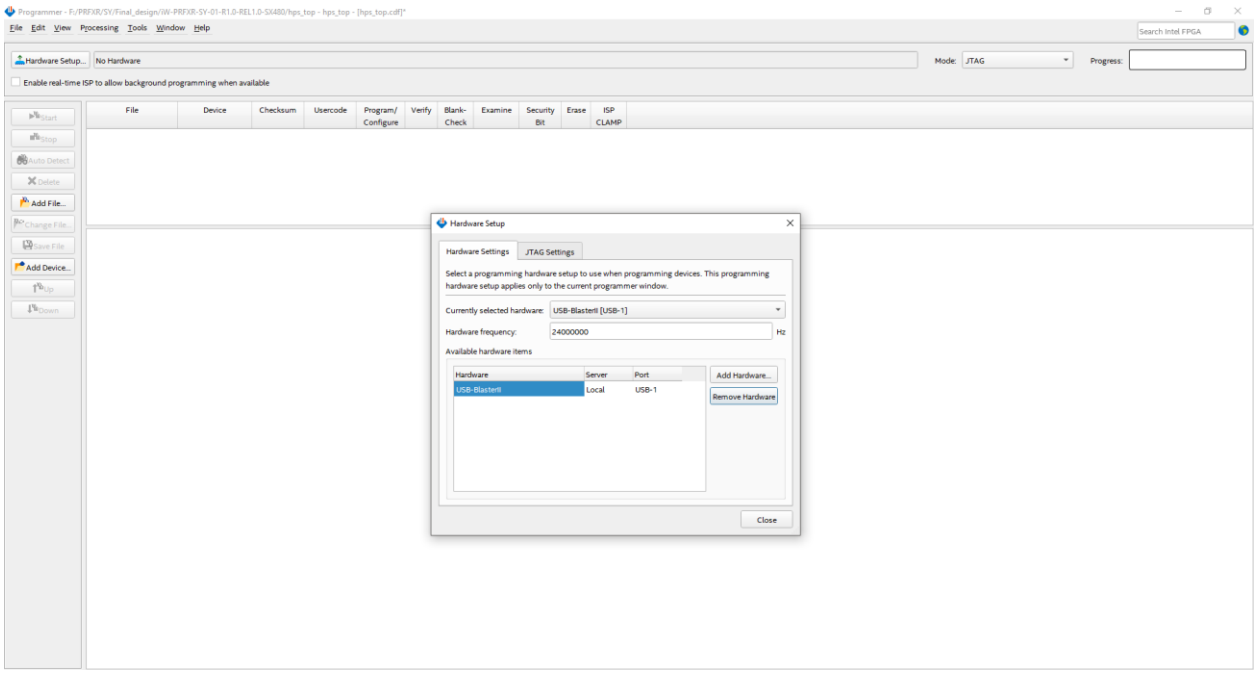


Figure 17: Programming Project - 4

- After JTAG is successfully detected, Select Auto Detect and select correct Part Number of the Arria10 SOC/FPGA SOM Board from the list of given Arria 10 Boards. Then click OK.

Note: Programming SX480 SOM, select the part number- (10AS048H3F34E2SG)

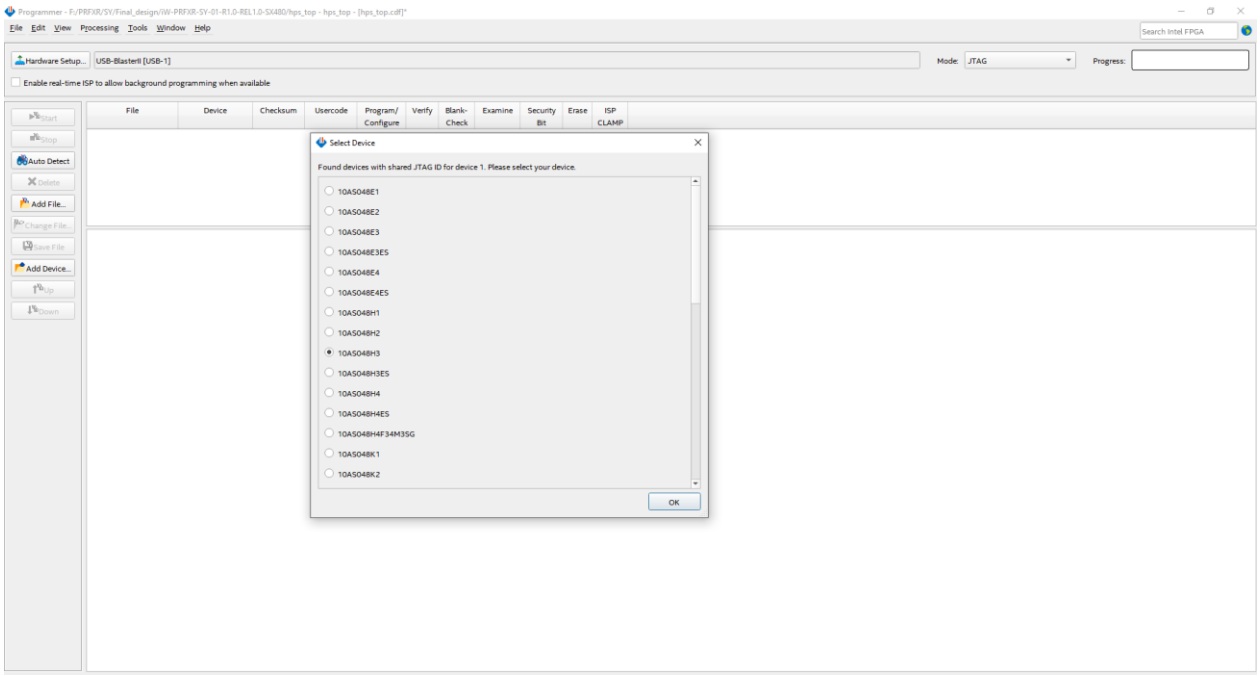


Figure 18: Programming Project - 5

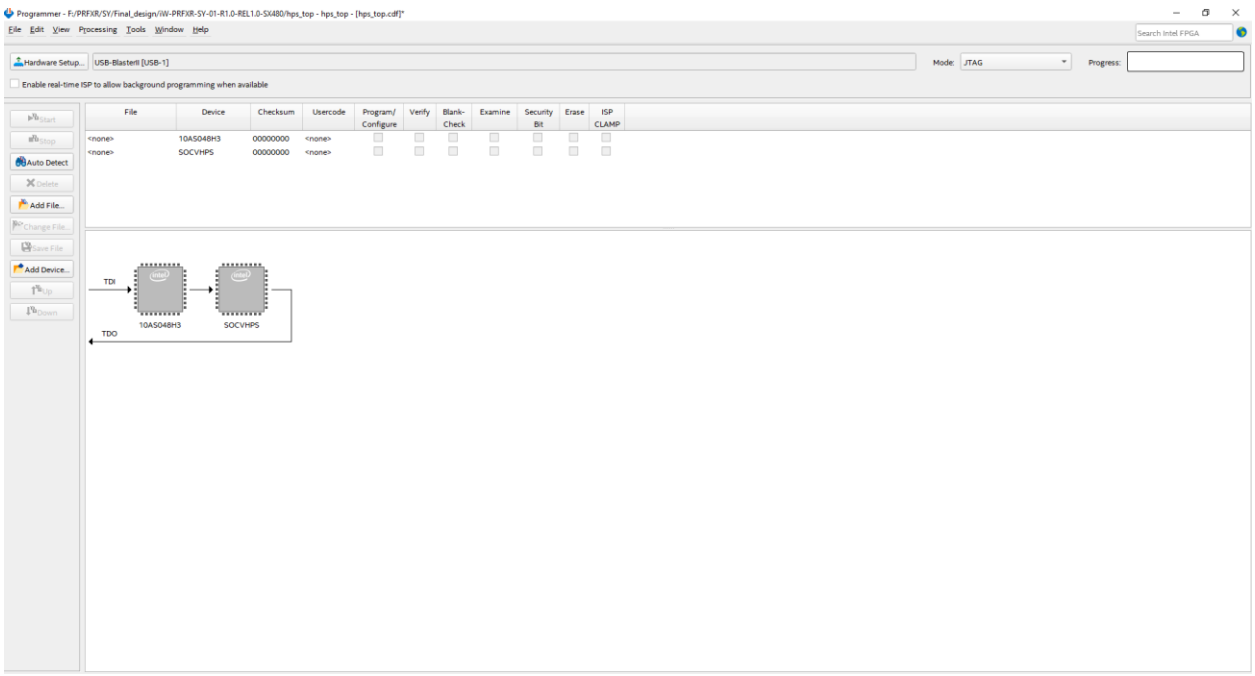


Figure 19: Programming Project - 5

- Select the Device and then Select the Add File option present at the left most side of the screen.

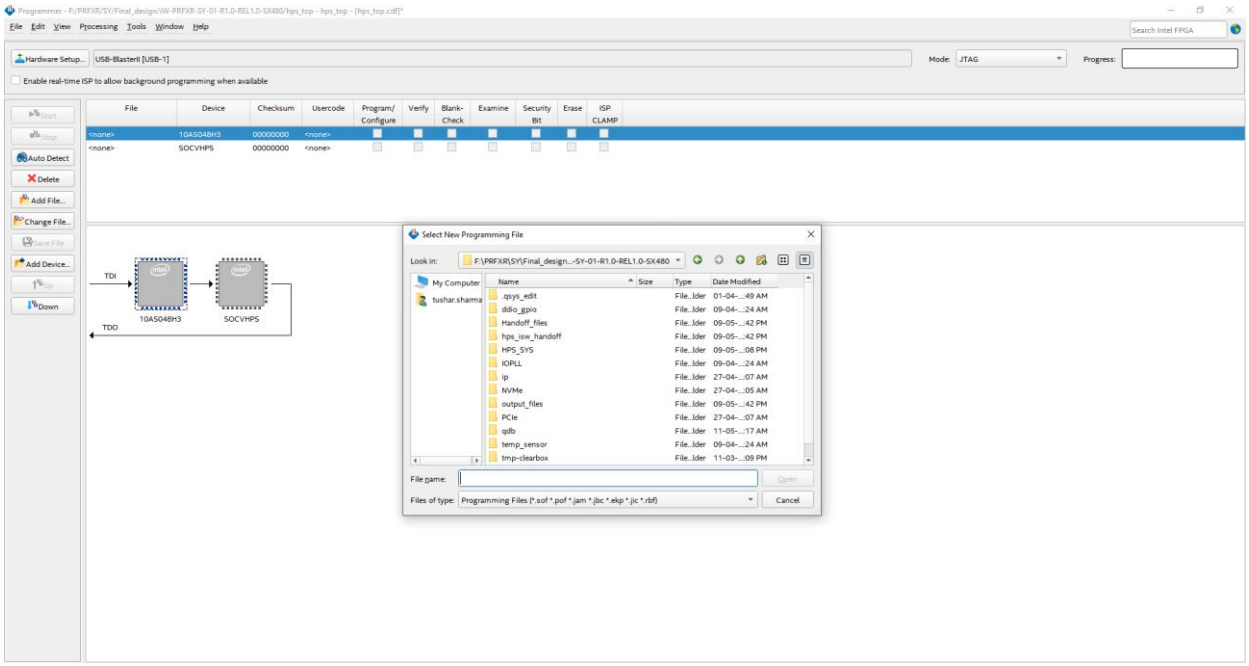


Figure 20: Programming Project - 6

- Select Programming File Dialogue Box will appear.

- In the Select Programming File Dialogue Box, select Output Files → hps_top.sof
- hps_top.sof is the binary file that is required for programming FPGA.

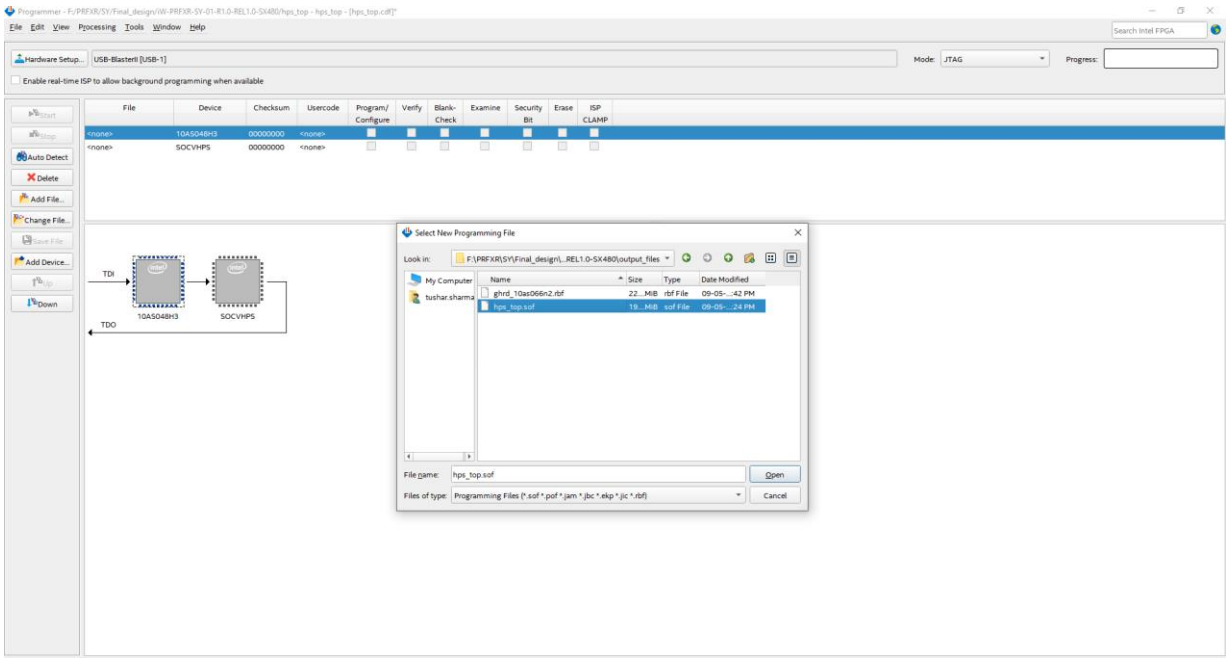


Figure 21: Programming Project - 7

- Once the file is loaded, Select the Program/Configure option and then Select Start.
- On the Top Right-hand side, the Progress bar will show 100%, giving us the information that our FPGA is successfully Programmed.

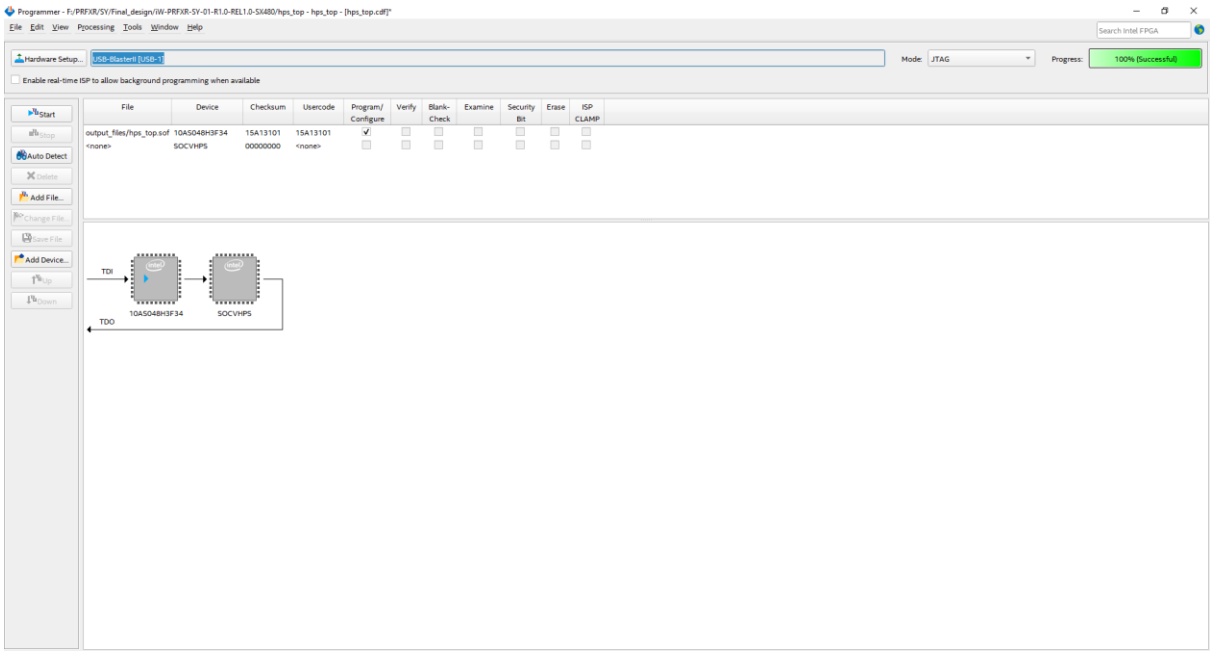


Figure 22: Programming Project - 8

- After the FPGA is successfully programmed, Minimize the Programmer.

4. Appendix

4.1 Transceiver Toolkit setup

- Refer section **FPGA Programming** to program the FPGA project using Quartus 18.1 prime pro tool.
- After Programming the FPGA project, Go to Tools → System Debugging Tools → Transceiver Toolkit.
- On clicking the Transceiver Toolkit option, the below page opens up.

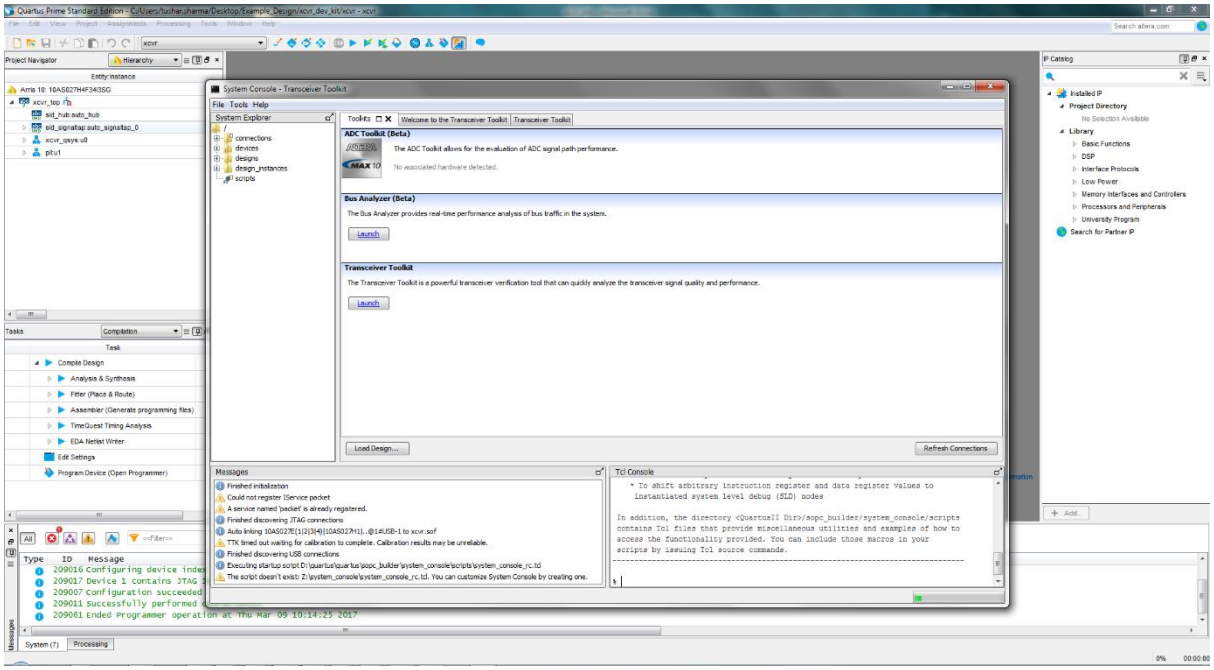


Figure 23: Startup Page for Transceiver Toolkit

- To load the design in the Transceiver Toolkit, Go to Files → Load Design.

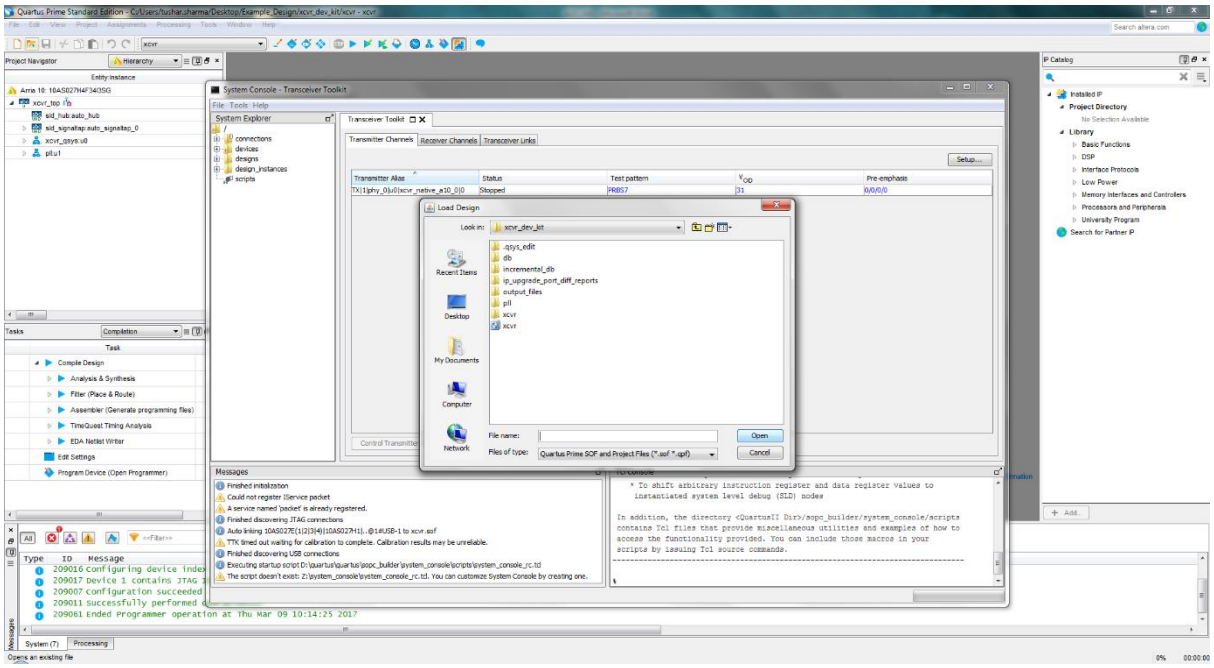


Figure 24: Loading Design into Transceiver Toolkit

- On selecting the load design option, the above Dialogue Box appears.Select the hps_top.sof file to load our design into the Toolkit.
- Once the file is loaded into our Toolkit, you can setup the Transceiver Toolkit with your own values.

- Under the Transceiver Toolkit Tab, Click the Setup Button.

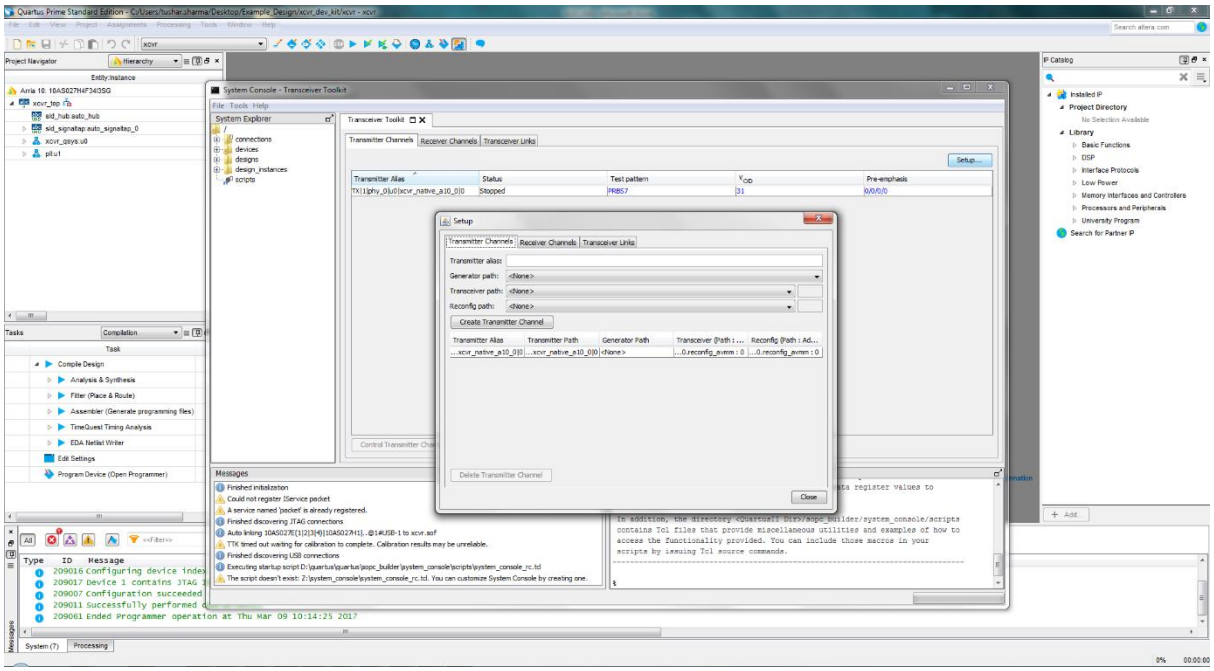


Figure 25: Setup Page of Transceiver Toolkit

- On clicking the Setup button, the above Dialogue Box appears.
- Before setting up your own values for the Transmitter Channel, Delete the previous values by selecting the present value and clicking the Delete Transmitter channel button.

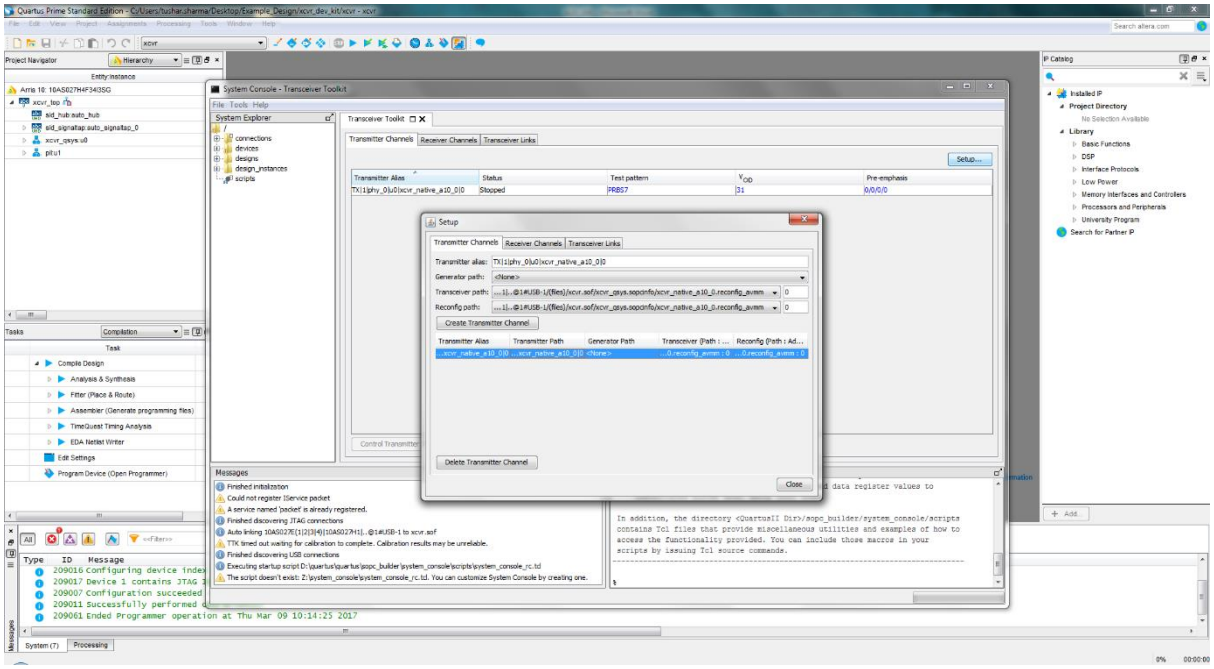


Figure 26: Configuring Transmitter Channel

- Now the previous values are deleted, Now create your own values by giving any name in the Transmitter Alias field (For eg. Transmitter or tx).
- Each channel will have its own Generator Path, Transceiver Path and Reconfig Path.
- For the example design, configure each channel by selecting respective Generator Path, Transceiver Path and Reconfig Path of each channel.
- For better understanding, refer the figures attached below to Setup your Toolkit.

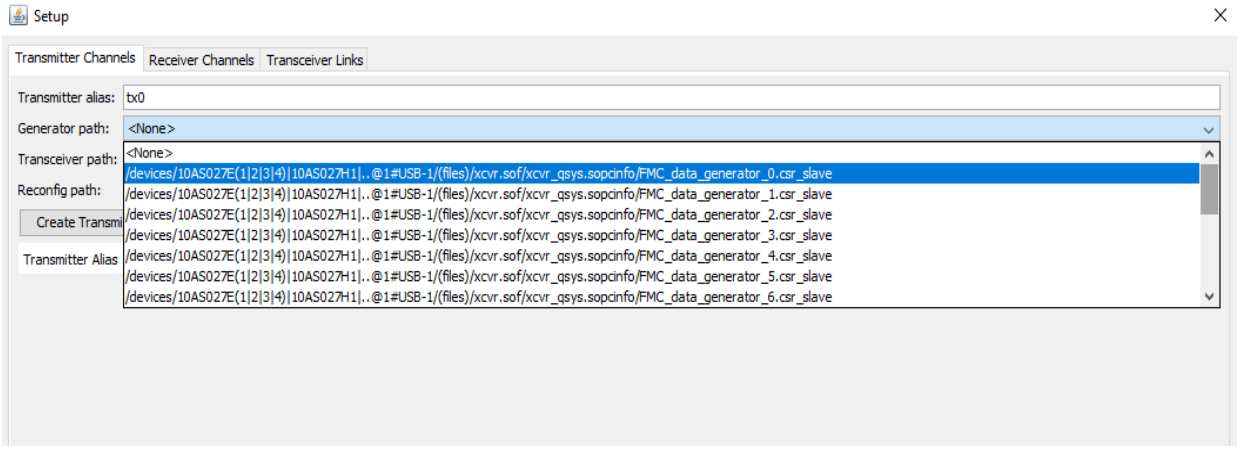


Figure 27: Configuring Generator path for FMC in Transmitter channel Tab

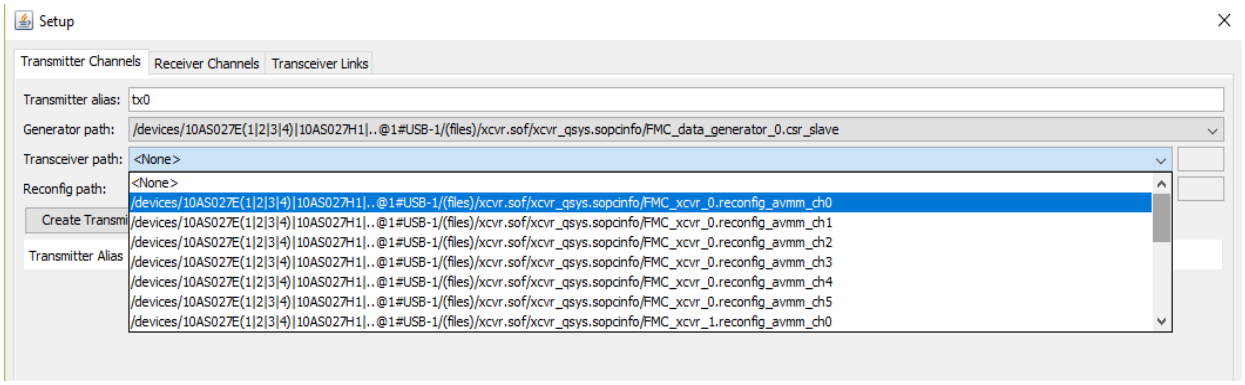


Figure 28: Configuring Transceiver path for FMC in Transmitter channel Tab

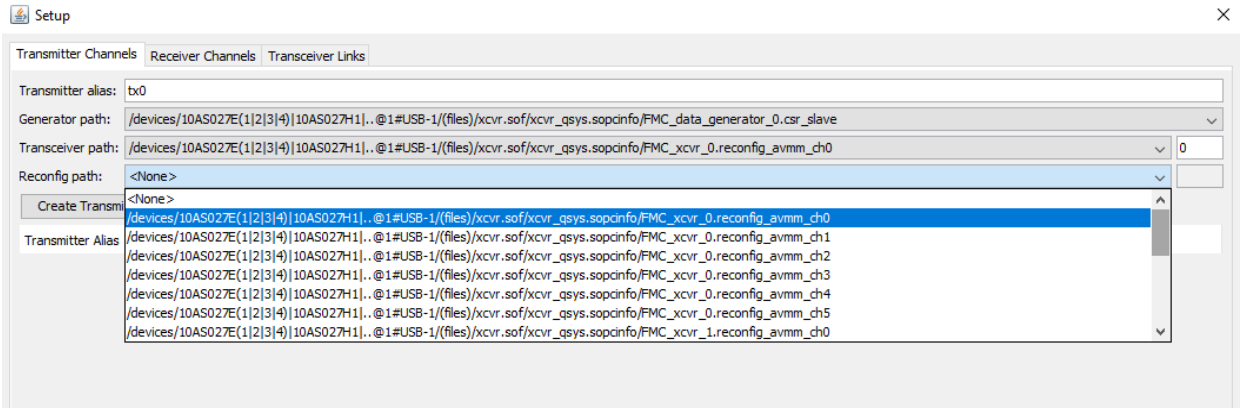


Figure 29: Configuring Reconfig path for FMC in Transmitter channel Tab

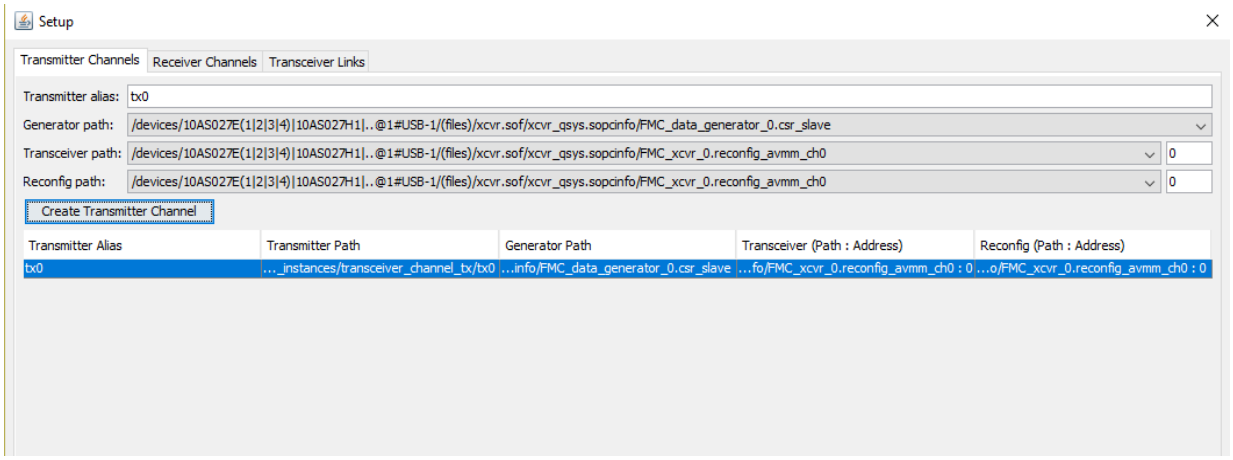


Figure 30: Configured FMC Transmitter channel

- After Selecting the values, Select the Create Transmitter Channel to create your Transmitter channel.
- Setup the Receiver Channel similar to Transmitter channel.

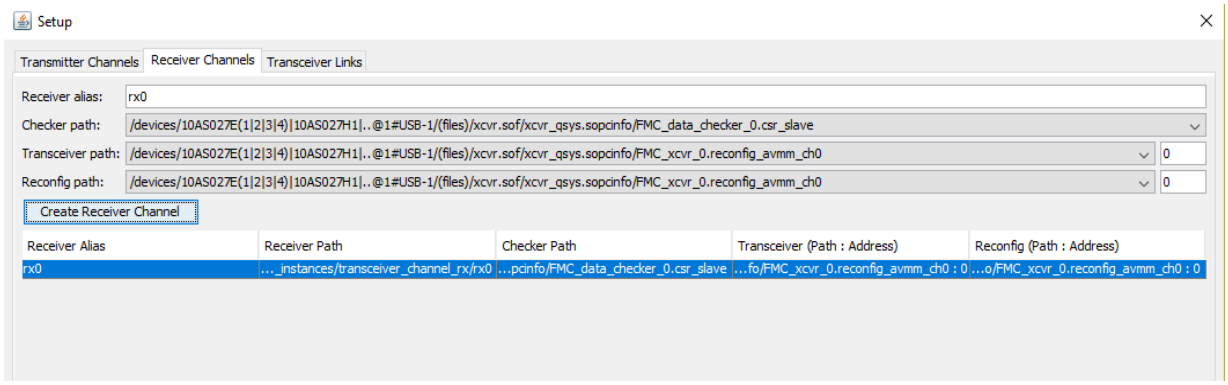


Figure 31: Configured FMC Receiver channel

- Setup the Transceiver Links similar to Transmitter channel.

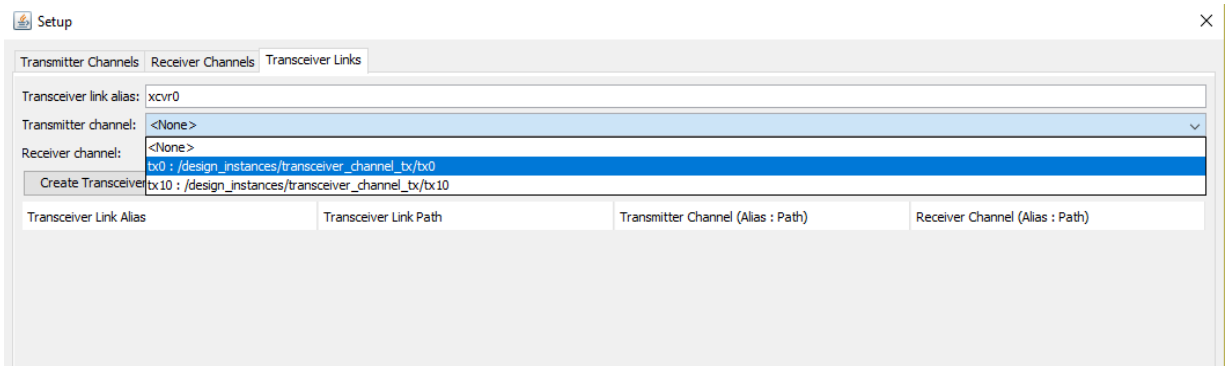


Figure 32: Configuring Transmitter Channel for FMC in Transceiver Link Tab

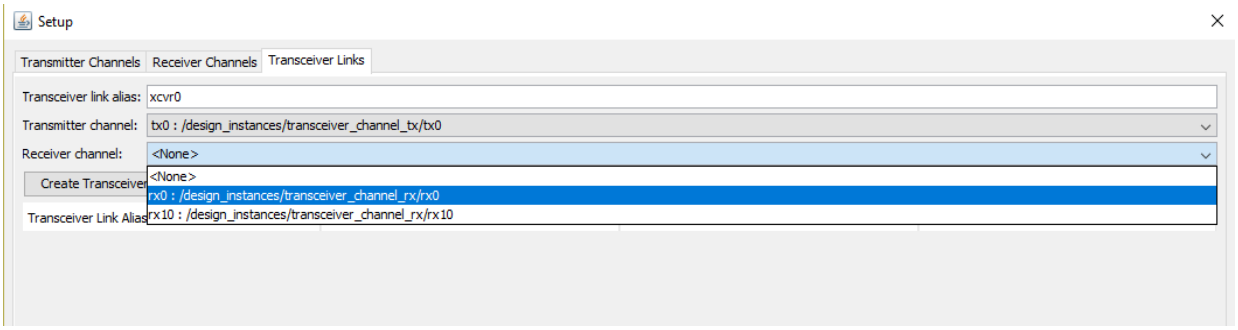


Figure 33: Configuring Receiver Channel for FMC in Transceiver Link Tab

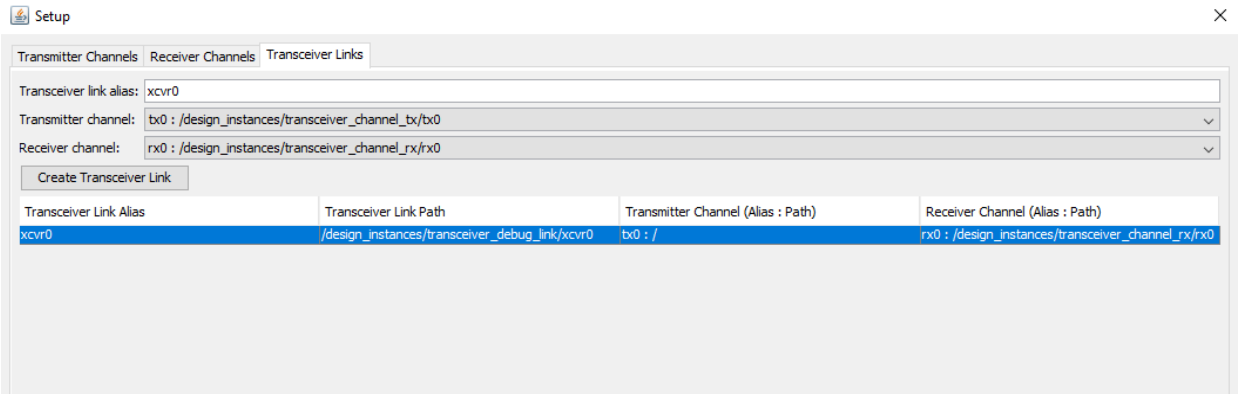


Figure 34: Configured FMC Transceiver Link

- Refer the Figure 35: Transciever Toolkit Interface Configuration list for the rest of the Signal details.

FMC INTERFACE										
TRANSMITTER CHANNEL				RECEIVER CHANNEL				Transceiver Link		
Transmitter Alias	Generator Path	Transceiver Path	Reconfig Path	Receiver Alias	Checker Path	Transceiver Path	Reconfig Path	Transceiver Link Alias	Transmitter Channel	Receiver Channel
tx0	FMC_data_generator_0	FMC_1C.reconfig_avmm_ch0	FMC_1C.reconfig_avmm_ch0	rx0	FMC_data_checker_0	FMC_1C.reconfig_avmm_ch0	FMC_1C.reconfig_avmm_ch0	xcvr0	tx0	rx0
tx1	FMC_data_generator_1	FMC_1C.reconfig_avmm_ch1	FMC_1C.reconfig_avmm_ch1	rx1	FMC_data_checker_1	FMC_1C.reconfig_avmm_ch1	FMC_1C.reconfig_avmm_ch1	xcvr1	tx1	rx1
tx2	FMC_data_generator_2	FMC_1C.reconfig_avmm_ch2	FMC_1C.reconfig_avmm_ch2	rx2	FMC_data_checker_2	FMC_1C.reconfig_avmm_ch2	FMC_1C.reconfig_avmm_ch2	xcvr2	tx2	rx2
tx3	FMC_data_generator_3	FMC_1C.reconfig_avmm_ch3	FMC_1C.reconfig_avmm_ch3	rx3	FMC_data_checker_3	FMC_1C.reconfig_avmm_ch3	FMC_1C.reconfig_avmm_ch3	xcvr3	tx3	rx3
tx4	FMC_data_generator_4	FMC_1D.reconfig_avmm_ch0	FMC_1D.reconfig_avmm_ch0	rx4	FMC_data_checker_4	FMC_1D.reconfig_avmm_ch0	FMC_1D.reconfig_avmm_ch0	xcvr4	tx4	rx4
tx5	FMC_data_generator_5	FMC_1D.reconfig_avmm_ch1	FMC_1D.reconfig_avmm_ch1	rx5	FMC_data_checker_5	FMC_1D.reconfig_avmm_ch1	FMC_1D.reconfig_avmm_ch1	xcvr5	tx5	rx5
tx6	FMC_data_generator_6	FMC_1D.reconfig_avmm_ch2	FMC_1D.reconfig_avmm_ch2	rx6	FMC_data_checker_6	FMC_1D.reconfig_avmm_ch2	FMC_1D.reconfig_avmm_ch2	xcvr6	tx6	rx6
tx7	FMC_data_generator_7	FMC_1D.reconfig_avmm_ch3	FMC_1D.reconfig_avmm_ch3	rx7	FMC_data_checker_7	FMC_1D.reconfig_avmm_ch3	FMC_1D.reconfig_avmm_ch3	xcvr7	tx7	rx7
tx8	FMC_data_generator_8	FMC_1E.reconfig_avmm_ch0	FMC_1E.reconfig_avmm_ch0	rx8	FMC_data_checker_8	FMC_1E.reconfig_avmm_ch0	FMC_1E.reconfig_avmm_ch0	xcvr8	tx8	rx8
tx9	FMC_data_generator_9	FMC_1E.reconfig_avmm_ch1	FMC_1E.reconfig_avmm_ch1	rx9	FMC_data_checker_9	FMC_1E.reconfig_avmm_ch1	FMC_1E.reconfig_avmm_ch1	xcvr9	tx9	rx9
tx10	FMC_data_generator_10	FMC_1E.reconfig_avmm_ch2	FMC_1E.reconfig_avmm_ch2	rx10	FMC_data_checker_10	FMC_1E.reconfig_avmm_ch2	FMC_1E.reconfig_avmm_ch2	xcvr10	tx10	rx10
tx11	FMC_data_generator_11	FMC_1E.reconfig_avmm_ch3	FMC_1E.reconfig_avmm_ch3	rx11	FMC_data_checker_11	FMC_1E.reconfig_avmm_ch3	FMC_1E.reconfig_avmm_ch3	xcvr11	tx11	rx11
tx12	FMC_data_generator_12	FMC_1F.reconfig_avmm_ch0	FMC_1F.reconfig_avmm_ch0	rx12	FMC_data_checker_12	FMC_1F.reconfig_avmm_ch0	FMC_1F.reconfig_avmm_ch0	xcvr12	tx12	rx12
tx13	FMC_data_generator_13	FMC_1F.reconfig_avmm_ch1	FMC_1F.reconfig_avmm_ch1	rx13	FMC_data_checker_13	FMC_1F.reconfig_avmm_ch1	FMC_1F.reconfig_avmm_ch1	xcvr13	tx13	rx13
tx14	FMC_data_generator_14	FMC_1F.reconfig_avmm_ch2	FMC_1F.reconfig_avmm_ch2	rx14	FMC_data_checker_14	FMC_1F.reconfig_avmm_ch2	FMC_1F.reconfig_avmm_ch2	xcvr14	tx14	rx14
tx15	FMC_data_generator_15	FMC_1F.reconfig_avmm_ch3	FMC_1F.reconfig_avmm_ch3	rx15	FMC_data_checker_15	FMC_1F.reconfig_avmm_ch3	FMC_1F.reconfig_avmm_ch3	xcvr15	tx15	rx15

Figure 35: Transciever Toolkit Interface Configuration list

- Under the Transceiver Link Tab, select the Transceiver value that you have created and click Control Transceiver Link Button.

Note: only 1 Transceiver channel is taken as example for further explanation.

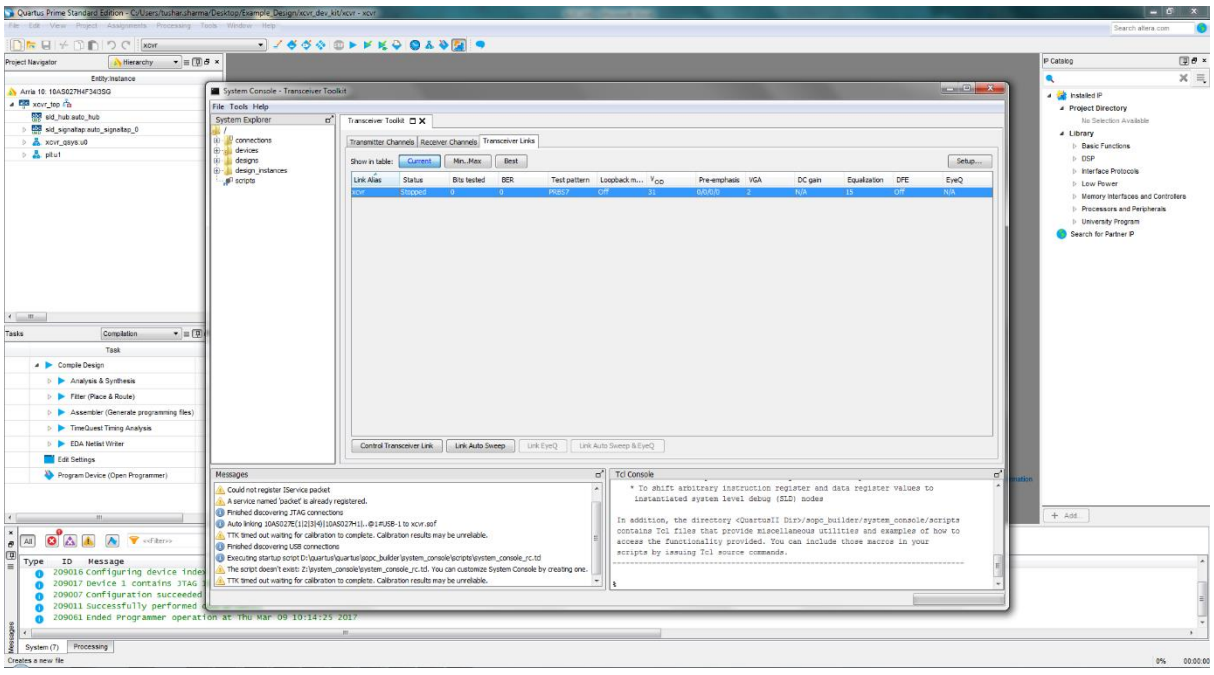


Figure 36: Controlling Transceiver Link

- On clicking the Control Transceiver Link, a new Transceiver Link Tab opens, as showed in the image below.

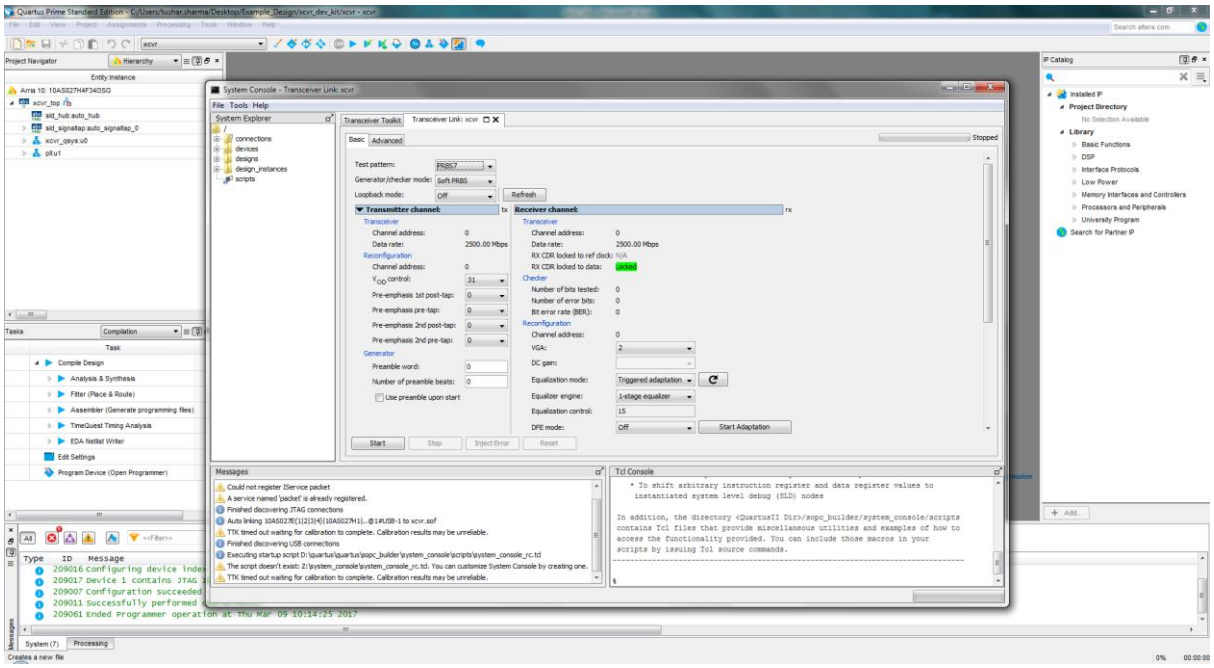


Figure 37: Basic Mode Startup Page

4.2 Transceiver Toolkit Test Procedure

Under Transceiver Link Tab, there are two options available

- Basic
- Advanced

Table 2: Settings for Transceiver Link

Link Control	Description
Test Pattern	Test pattern sent by the Transmitter channel. Options include: • PRBS 7 • PRBS 15 • PRBS 23 • PRBS 31 • Low Frequency • High Frequency The Data Pattern Checker self-aligns both high and low frequency pattern.
Generator/Checker Mode	Soft PRBS
Loopback Mode	Loopback Mode options include: • OFF • Metallic • Reverse Serial • Serial Loopback *Note: Use OFF option.*
Channel Address	Logical Address numbers of the Transceiver channel
Data Rate	Data Rate of the channel as read from the project file or data rate as measured by the frequency detector. To use the frequency detector, turn on Enable Frequency Counter in the Data Pattern Checker IP core and/or Data Pattern Generator IP core, regenerate the IP cores, and recompile the design. Click the Refresh button next to the measured Data rate if you make changes to your settings and what to sample the data rate again.
VOD Control	Programmable transmitter differential output voltage.
Pre-emphasis 1st post-tap, Pre-emphasis pre-tap, and Pre-emphasis 2nd post-tap	The programmable pre-emphasis module in each transmit buffer boosts high frequencies in the transmit data signal, which may be attenuated in the transmission media. Using pre-emphasis can maximize the data eye opening at the far-end receiver.

Link Control	Description
Preamble word	Word to send out if the preamble mode is used.
Number of preamble beats	The number of clock cycles to which the preamble word is sent before the test pattern begins.
RX CDR Locked to reference clock	Shows the receiver in lock-to-reference (LTR) mode. When in auto-mode, if data cannot be locked, this signal alternates in LTD mode if the CDR is locked to data.
RX CDR locked to data	Shows the receiver in lock-to-data (LTD) mode. When in auto-mode, if data cannot be locked, the signal stays high when locked to data and never toggles.
Number of bits tested	Specifies the number of bits tested since the last reset of the checker.
Number of error bits	Specifies the number of error bits encountered since the last reset of the checker.
Bit error rate (BER)	Specifies errors divided by bits tested since the last reset of the checker.
DC gain.	Circuitry that provides an equal boost to the incoming signal across the frequency spectrum.
Equalization control	Boosts the high-frequency gain of the incoming signal, thereby compensating for the low-pass filter effects of the physical medium. AEQ one-time adaptation is supported in Auto Sweep. When used with DFE, you need to use DFE triggered mode or DFE continuous.
DFE 1st tap value, DFE 2nd tap value, DFE 3rd tap value, DFE 4th tap value, DFE 5th tap value	Decision feedback equalization (DFE) for improving signal quality. 0=off, 1=adaptive, 2= one-time adaptive. One-time mode DFE determines the best tap settings and stops searching. There is also a one-time adaptive mode button that automatically turns on one-time mode and immediately populates converged values into the manual settings lists. Adaptive mode DFE automatically tries to find the best tap values.
Start	Starts the pattern generator or checker on the channel to verify incoming data.
Stop	Stops generating patterns you have set for a channel.
Inject Error	Flips one bit to the output of the data pattern generator to introduce an artificial error.
Reset	Resets the current test.

4.2.1 Basic Mode

- Once your setup is ready for testing, First and foremost thing to look is that the Receiver Clock Data Recovery (Rx CDR) signal is Locked to your incoming Data i.e RX CDR Locked to Data should show Locked, as shown in the below Figure.

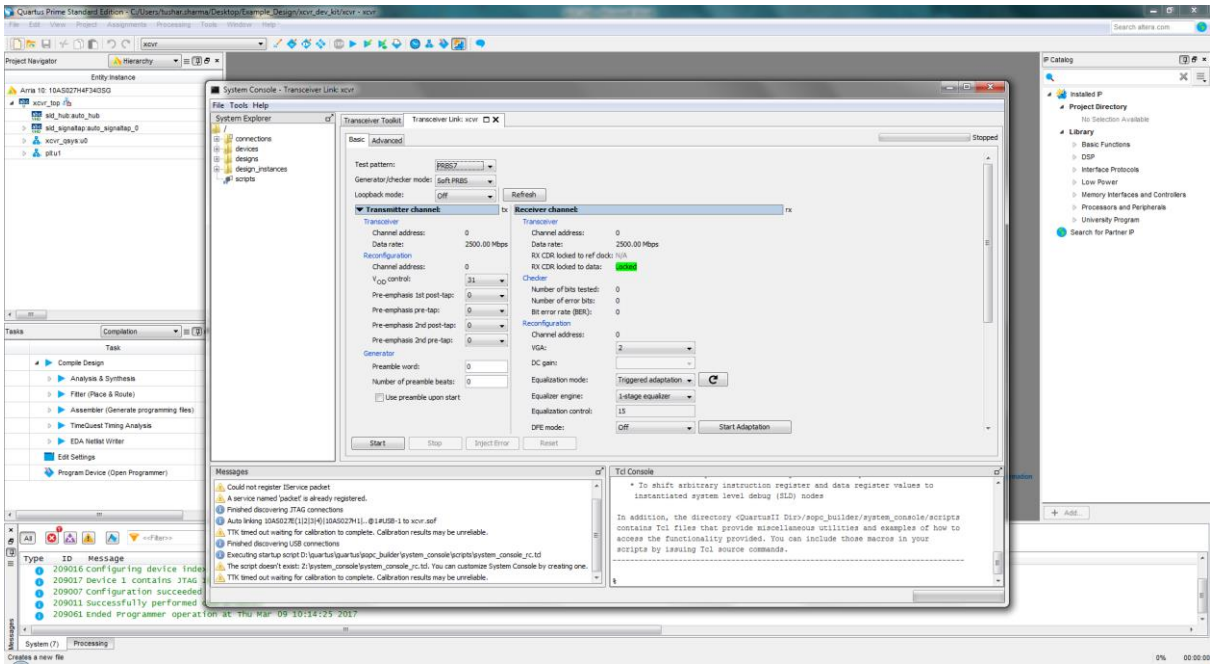


Figure 38: Testing Transceiver Toolkit

- Select the width of your data to be checked from the given Test Pattern options.
- Now start your testing by clicking on the Start button. After your testing has started, look for the color on the sides of the Transmitter Channel and Receiver Channel which should tell the status of your ongoing testing.

Table 3: Color depicting current process

Colour	Transmitter Channel	Receiver Channel
Red	Channel is closed (or) Generator clock is not running.	Channel is closed (or) Checker clock is not running.
Green	Generator is sending a pattern.	Checker is checking and data pattern is locked.
Neutral	Channel is open, generator clock is running, and generator is not sending a pattern (or) Generator is not sending a pattern.	Channel is open, checker clock is running, and checker is not checking (or) Checker is not checking.
Yellow	N/A	Checker is checking and data pattern is not locked.

- In this window, Number of Tested bits will increase indicating the number of bits being tested.
- Number of error bits should remain Zero indicating that there is no loss of data.
- Bit error rate should remain Zero indicating that there is no loss of data.

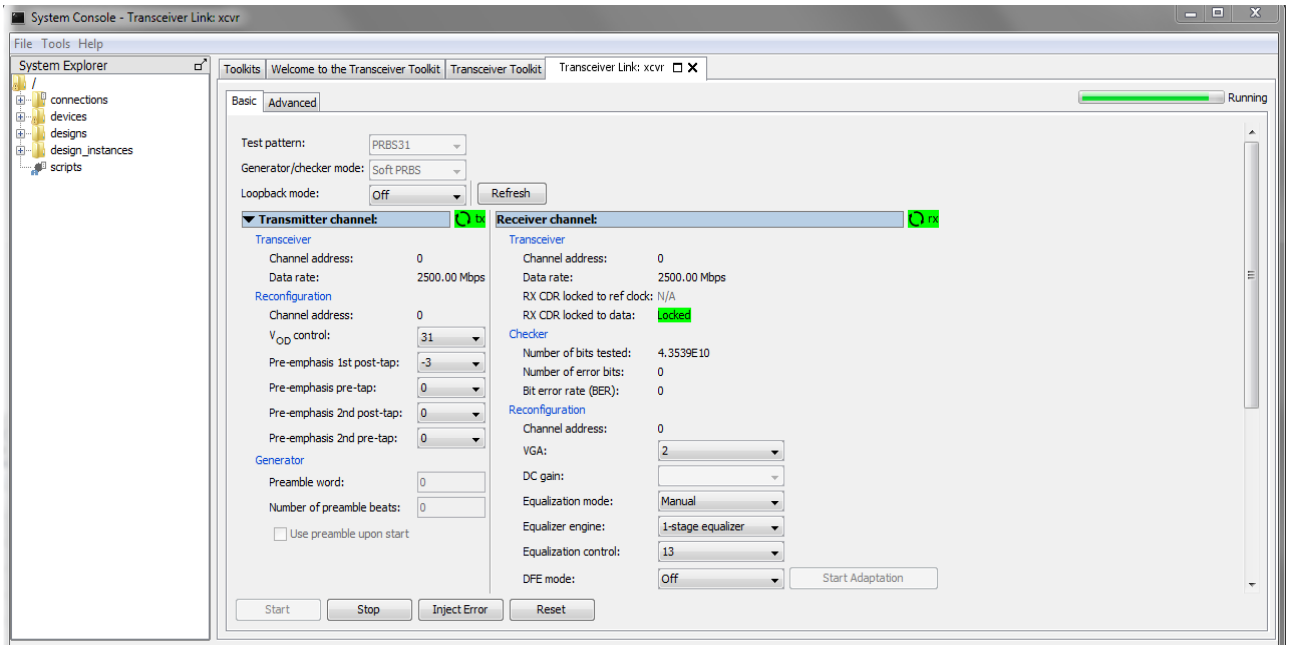


Figure 39: Checking Transceiver Toolkit value

- If the value of Number of error bits and Bit error rate increases from zero then you need to change the values of the above-mentioned parameters to make it error free.

Note: If no values are found in the Basic Mode, then you have to switch to Advanced Mode for finding the appropriate values for your transceiver to work to provide error free values.

4.2.2 Advanced Mode

- Once your setup is ready for testing, In the Toolkit window next Tab is the Advanced tab, when Advanced Tab is selected below page appears.

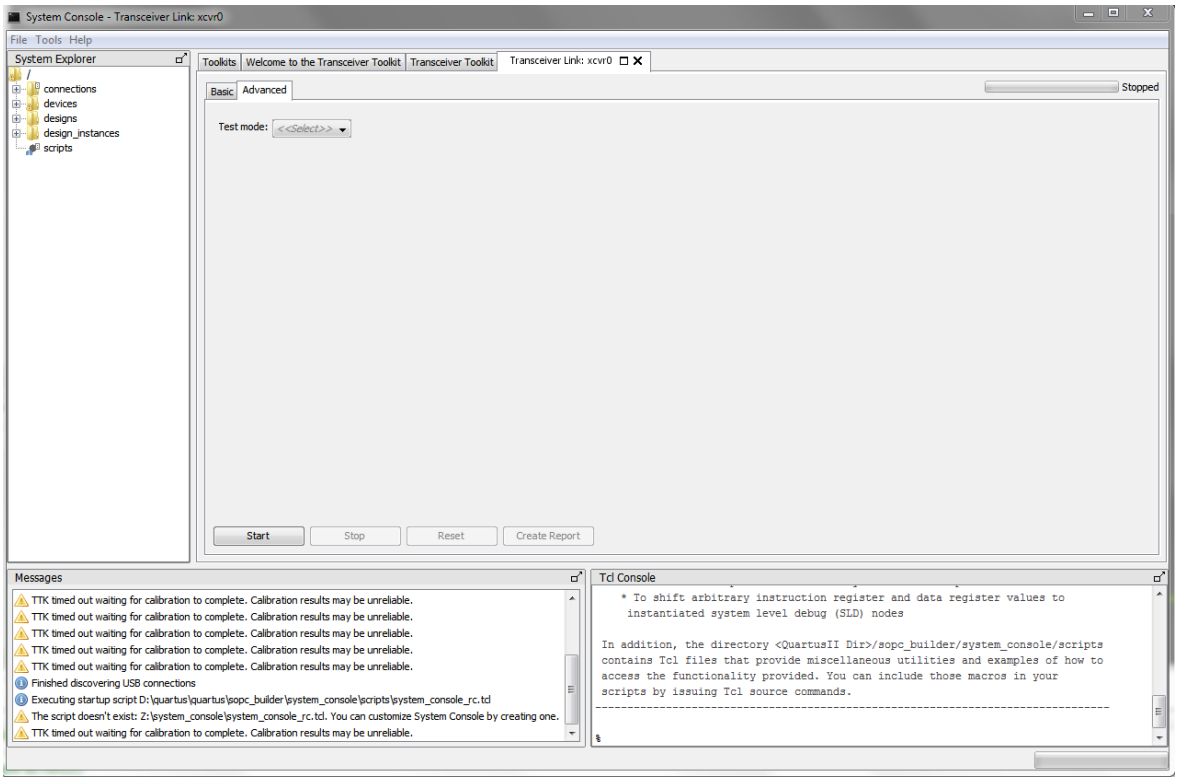


Figure 40: Advanced Mode Startup Page

- Select the Test Mode to Auto Sweep.

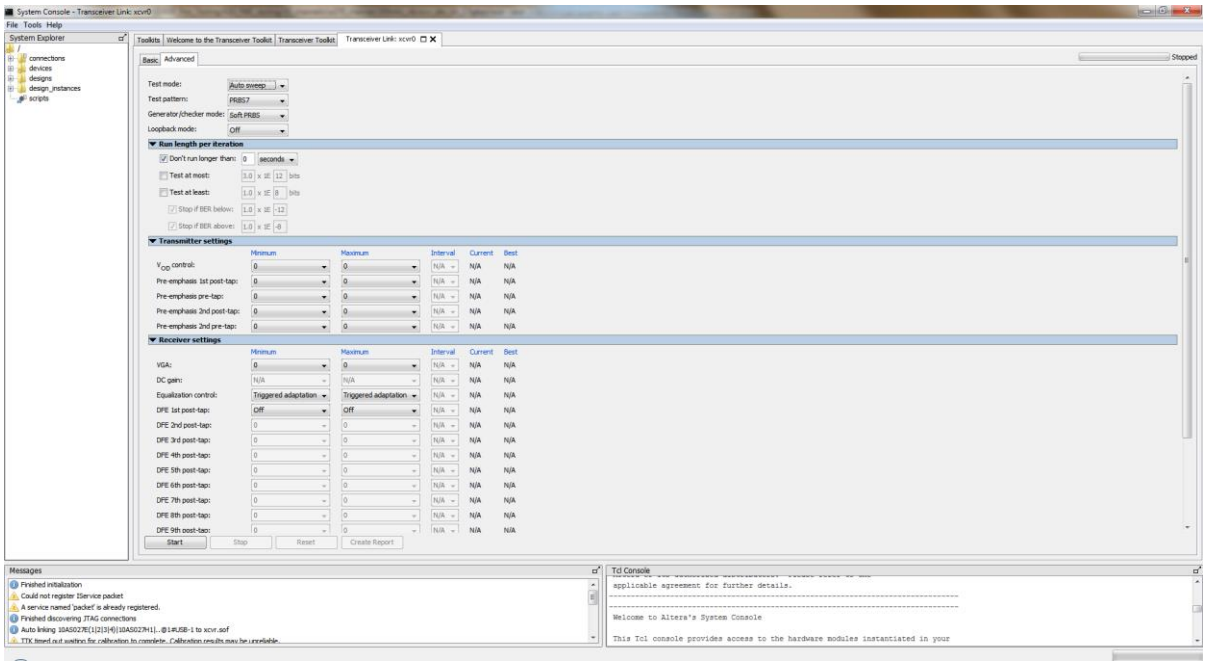


Figure 41: Advanced Mode Parameter Page

- In this Tab, you can select a range of values you want to run your toolkit.
- Don't Run Longer Than: Here you can set the amount of time you want your tests to be ran for.

Table 4: Analog Parameter Range

Colour	Range	
	Minimum	Maximum
Vod	30	31
Pre-emphasis 1st Post tap	-12	-4
Pre-emphasis 1st Pre-Tap	-12	-4
Pre-emphasis 2nd Post tap	-3	3
Pre-emphasis 2nd Pre tap	-3	3
VGA	2	5
Equalization Control	14	15

- An example of this above table can be seen in the below figure.

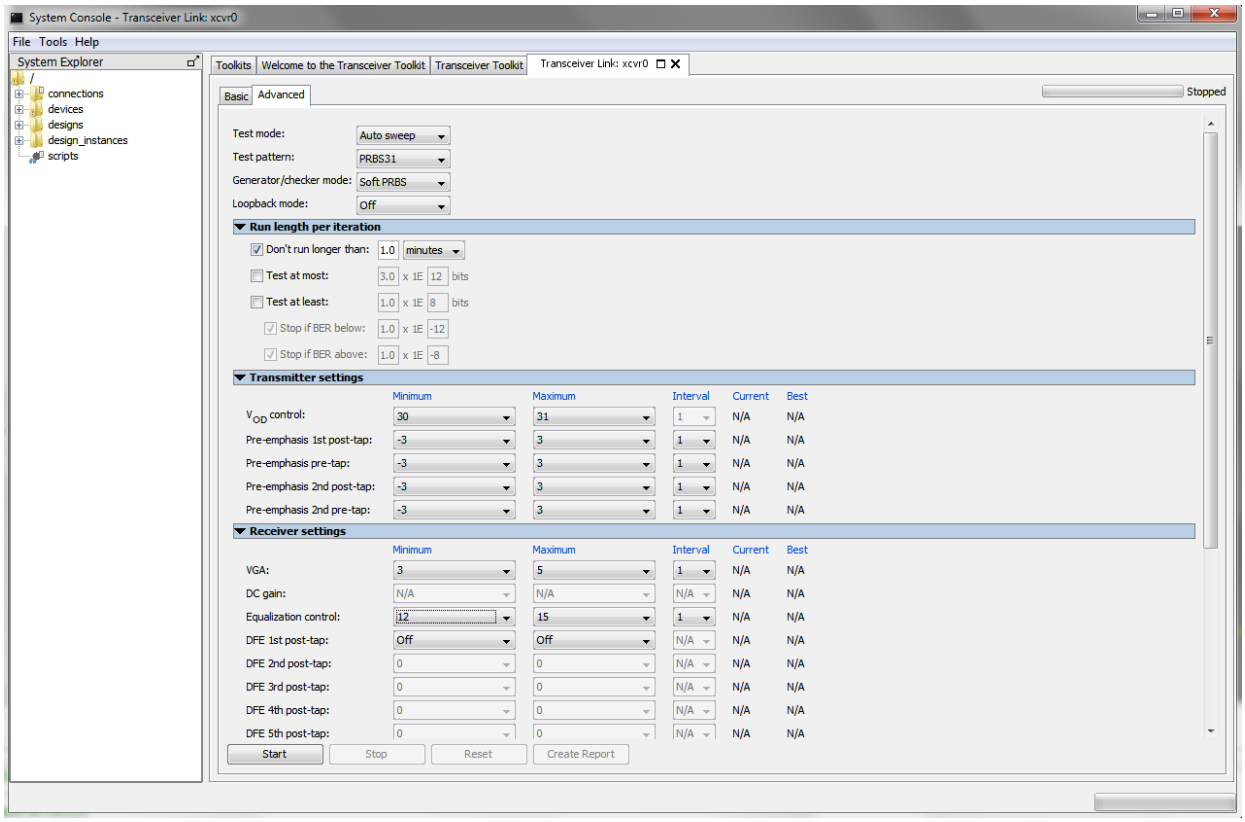


Figure 42: Configuring Advanced Mode

- Once the values are set, start your testing by clicking the Start button. The observation can be seen in the below figure.

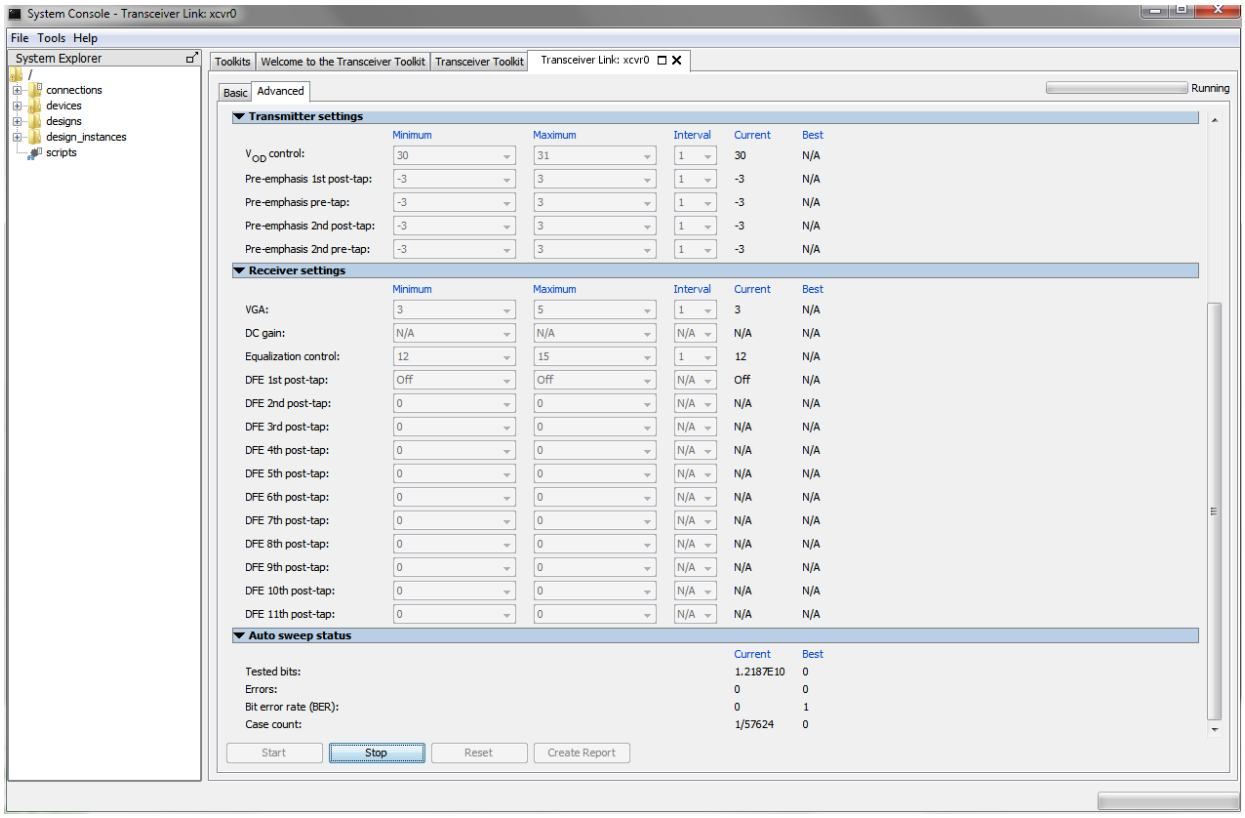


Figure 43: Testing Transceiver Toolkit

- The number of Tested Bits will increase but the values of Errors and Bit Error Rate should remain Zero.
- This Auto Sweep Mode will continuously check for the given values between ranges and will provide the best Test case scenarios for us.

- By setting the values in the provided range you'll get approximately 1 lakh test cases which should give you particular values where the Errors and Bit Error Rate would be Zero, as shown in the Figure below.

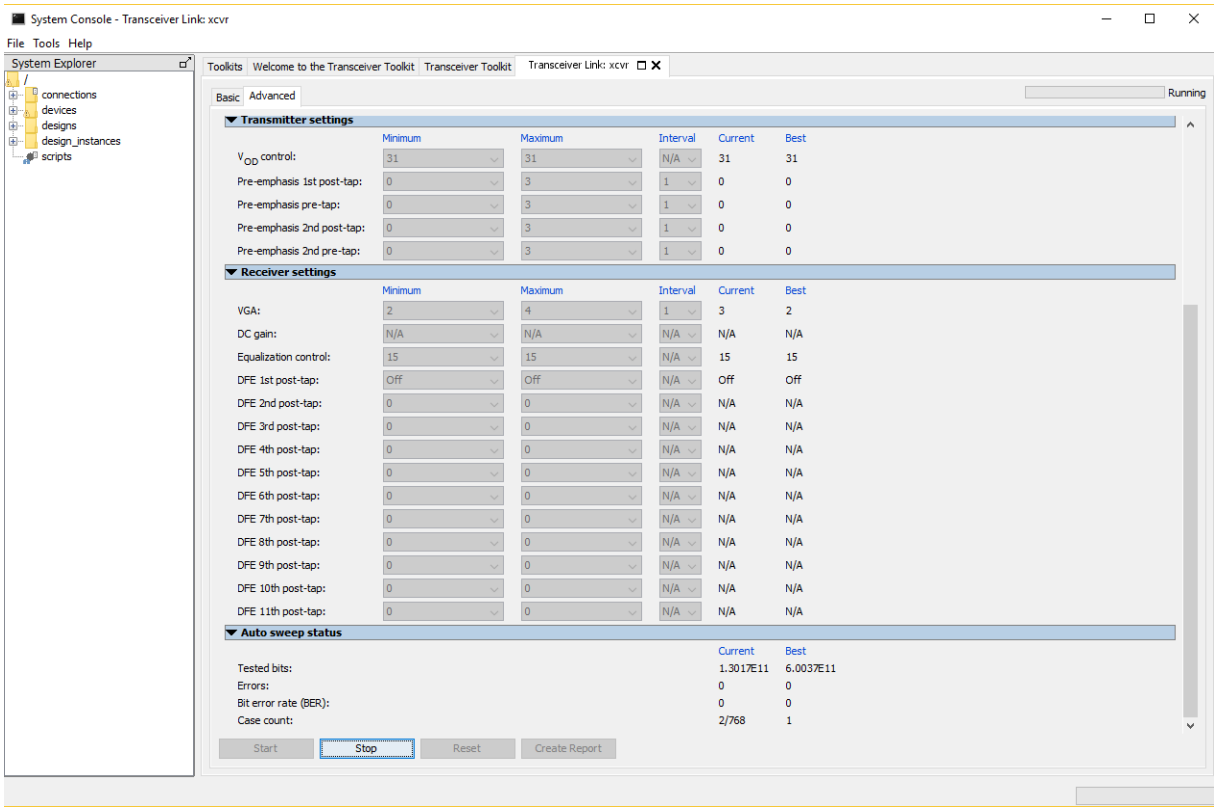


Figure 44: Best Value obtained


iWave
Embedding Intelligence