ZCU102 ADC12DJ1350 JESD REFERENCE DESIGN USER GUIDE

Table of Contents

I. DESCRIPTION

The design "ZCU102_ADC12DJ1350_8G.zip" is developed for ZCU102 board (HW-Z1-ZCU102, Revision D2 PROD) for the mode: JMODE0. It has JESD Base IP and JESD PHY IP to get JESD data from the ADC12DJ1350 and is compiled for 8G lane rate. This project is compiled for the part number XCZU9EG-2FFVB1156I. The design has a simple transport layer specific for the mode that captures samples from the ADC, re-order the bits and give out 10 samples every clock cycle. The results can be verified using Chipscope. This document gives a brief on the compilation and verification process involved.

NOTE: This version of the FW is a fixed line rate firmware and hence will work only at 8G. For any other line rate the firmware needs to be recompiled for that specific linerate.

Section II discusses about how the hardware setup should be connected. Section III discusses on how to extract the project from .zip file and the compilation process involved. Section IV discusses on how to get the ADC data in Chipscope and validate the same. Section V discusses the LED debug signals added in project.

Signals	Description	Direction
CLK_IN1_D1	User programmable clock. It's default value 300Mhz. However it is recommended to set to 300Mhz again through System controller.	Input
refclk	Reference clock from the ADC12DJ1350 EVM	
sysref	SYSREF Signal from the 12DJ1350 EVM	
Rxp_in/Rxn_in	Serial Data from ADC (LVDS lines)	
rx_dataout (Transport_smpl in chipscope)	Transport Layer data out (10 samples for every link clock)	Output from the transport module
rx_validout	SOMF aligned with rx_dataout	
rx_somfout	Data valid Signal aligned with rx_dataout	

ADC12DJ1350 EVM RevE1 was used for testing. In this EVM revision P & N pins of the four SERDES lanes are inverted, hence Rx lane polarity inversion is implemented in the design to address that. Due to this, Rx lane polarity inversion constant given to the PHY module is 15.

II. HARDWARE SETUP

Connect the ZCU102 board with 12DJ1350. Please use the **HPC0** FMC to connect the devices. Connect the Digilent port or the JTAG cable to the PC in order to download the firmware. Additionally UART port must also be connected to the PC to make any configurations.

III. COMPILING VIVADO PROJECT

1. Extracting the zip file.

- Right click the ZCU102_12DJ1350_8G.xpr.zip file and then press “Extract All”. After extracting, open the project using **Vivado 2017.2**. The project file is ZCU102_12DJ1350_8G.xpr\prj_MyZCU102_TI\ prj_MyZCU102_TI.xpr

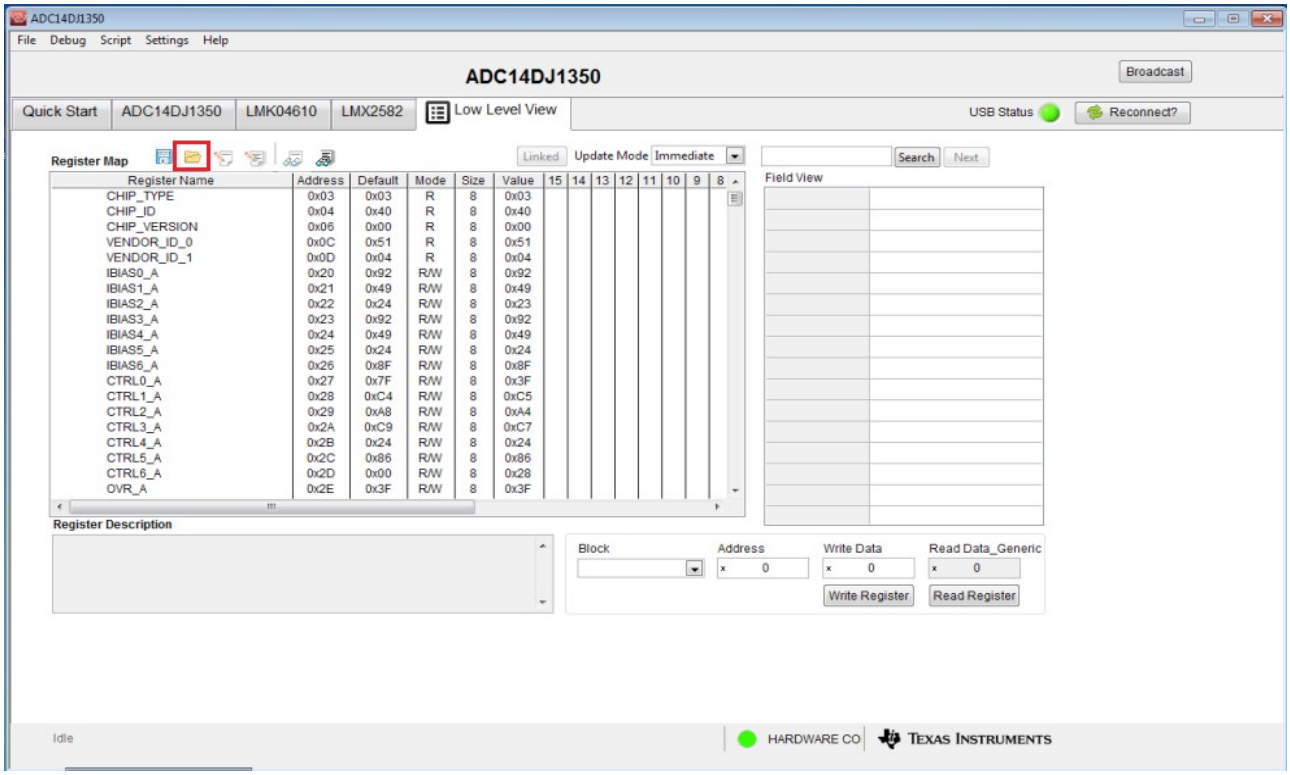
2. Compiling the project.

- In Vivado, on the left you will see the Flow Navigator. Press the generate Bitstream option under “Program and Debug” to generate the output “.bit” file.
- Once the generation is over a .bit file will be generated, along with with .ltx file (required for chipscope).
- The bit file generated is “mySystem_wrapper.bit”. The name of the .ltx file is debug_nets.ltx. These are the files that must be loaded when the device is to be programmed.
- These files are generated at the location - ZCU102_12DJ1350_8G.xpr\prj_MyZCU102_TI\prj_MyZCU102_TI.runs\impl_1\

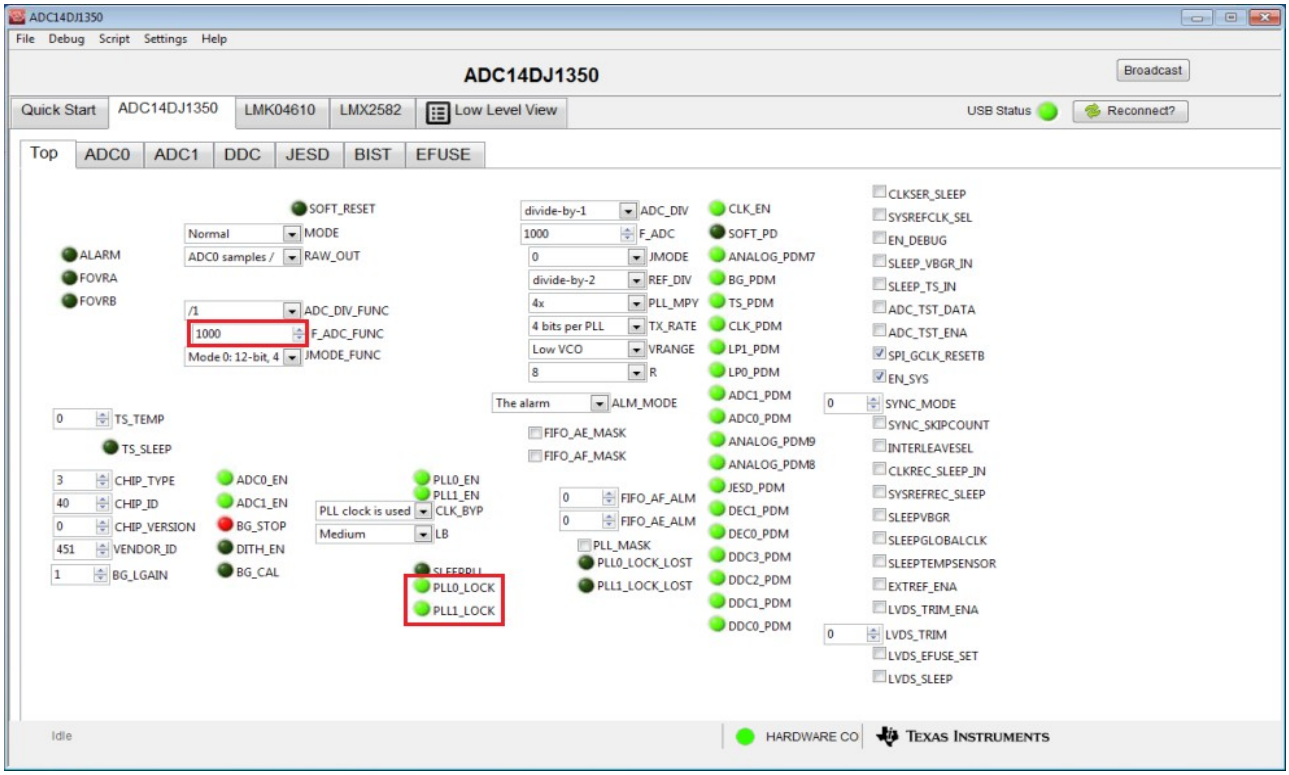
IV. VALIDATING WITH ADC12DJ1350

1. Configure ADC

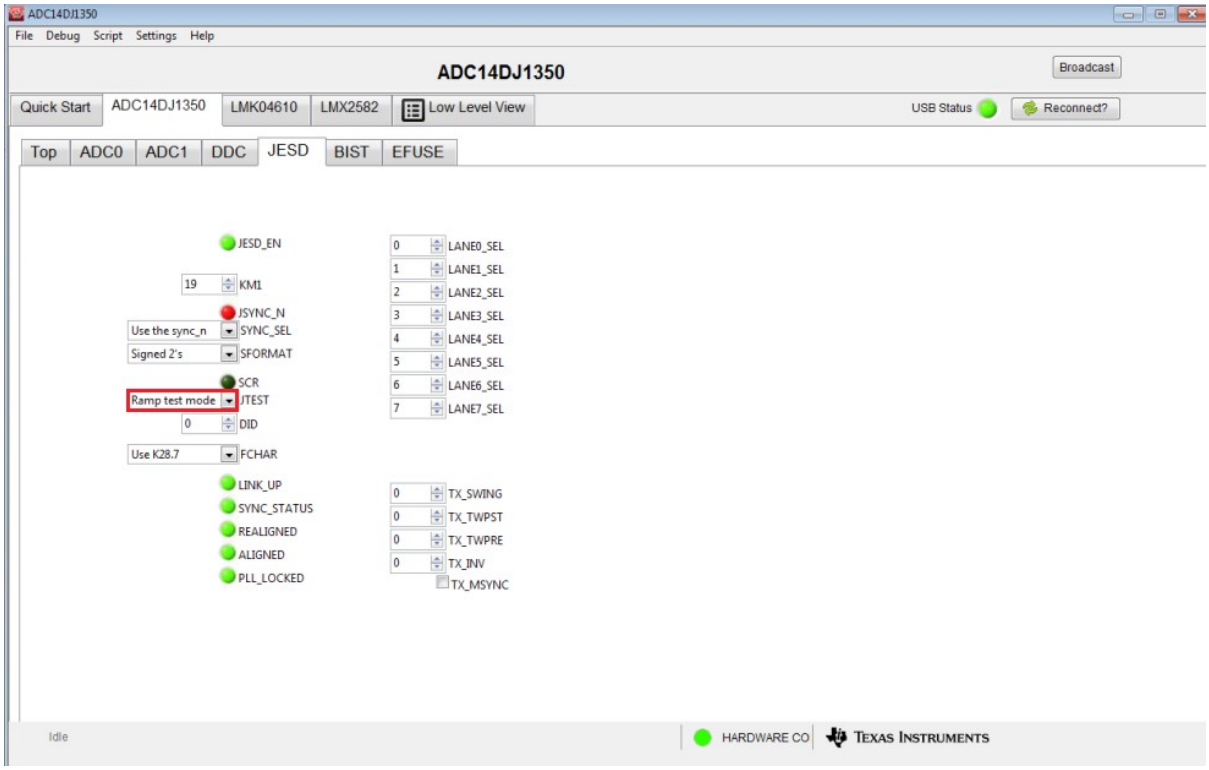
- The reference clock from ADC should be stable before downloading firmware. So, the ADC has to be configured first before we program the board.
- Open ADC12DJ1350 GUI, and manually download the following files one-by-one in the same order as mentioned by pressing the folder icon on the “Low level view” Tab.
    - LMK04610_MODE0_CLKIN1.cfg
    - LMX_1000M.cfg
    - ADC12DJ1350_Mode0.cfg (present within ADC12DJ1350 folder)



- Once it is successful, go to ADC14DJ1350 tab, and in the “Top” subtab type in 1000 for the “F_ADC_FUNC” variable. After doing this, please click on the PLL0_lock and PLL1_lock signals below on the same page to make sue the clock has been locked. (This step can be skipped if the normal sine wave input signal is to be captured).



- Finally go the JESD subtab and set the “JTest” variable to **Ramp test mode.** This will enable the device to send a separate byte ramp on each lane.(This step can be skipped if the normal sine wave input signal is to be captured).

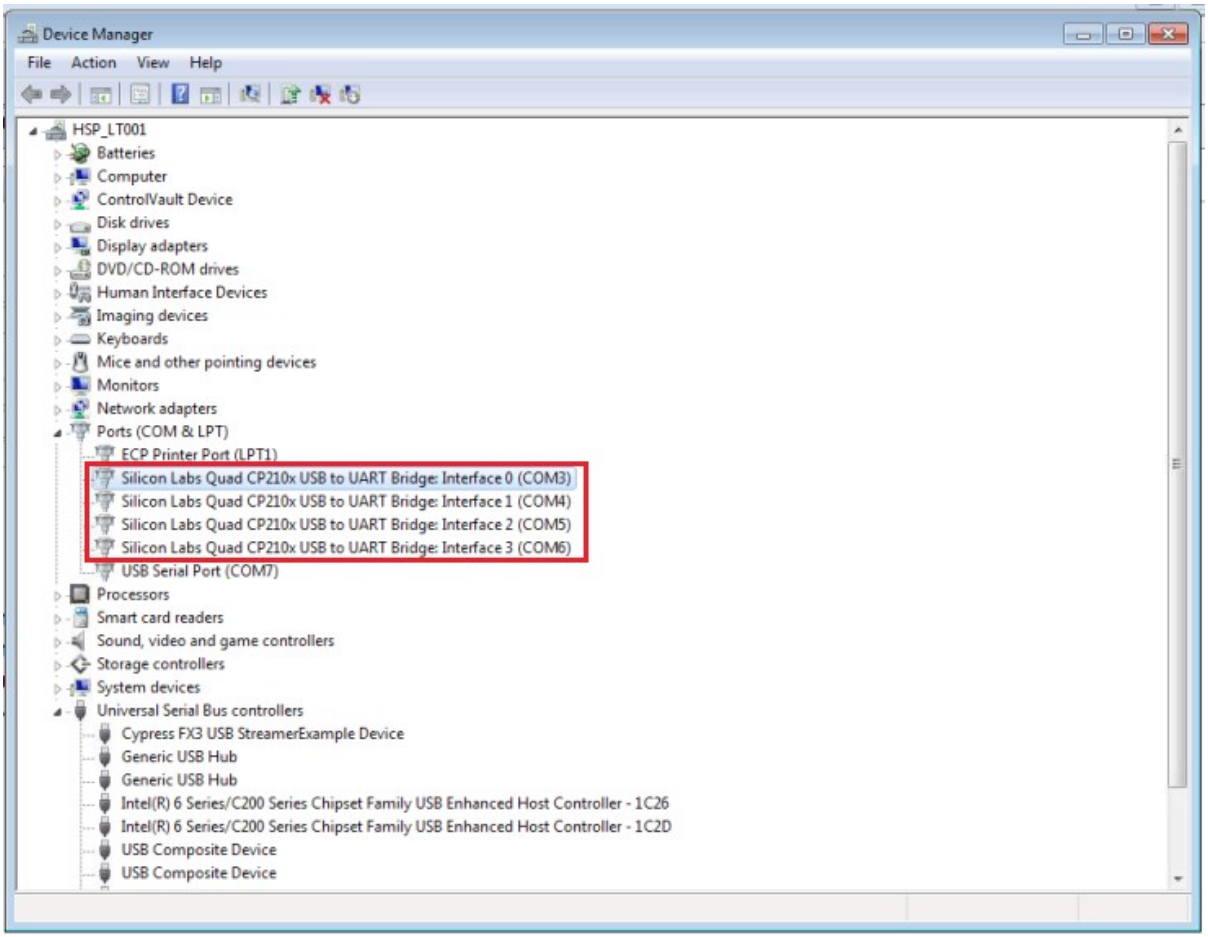


2. Programming with Vivado Hardware Manager

In order to program the ZCU102 we need to follow a series of steps. Please make sure you follow the order while programming it.

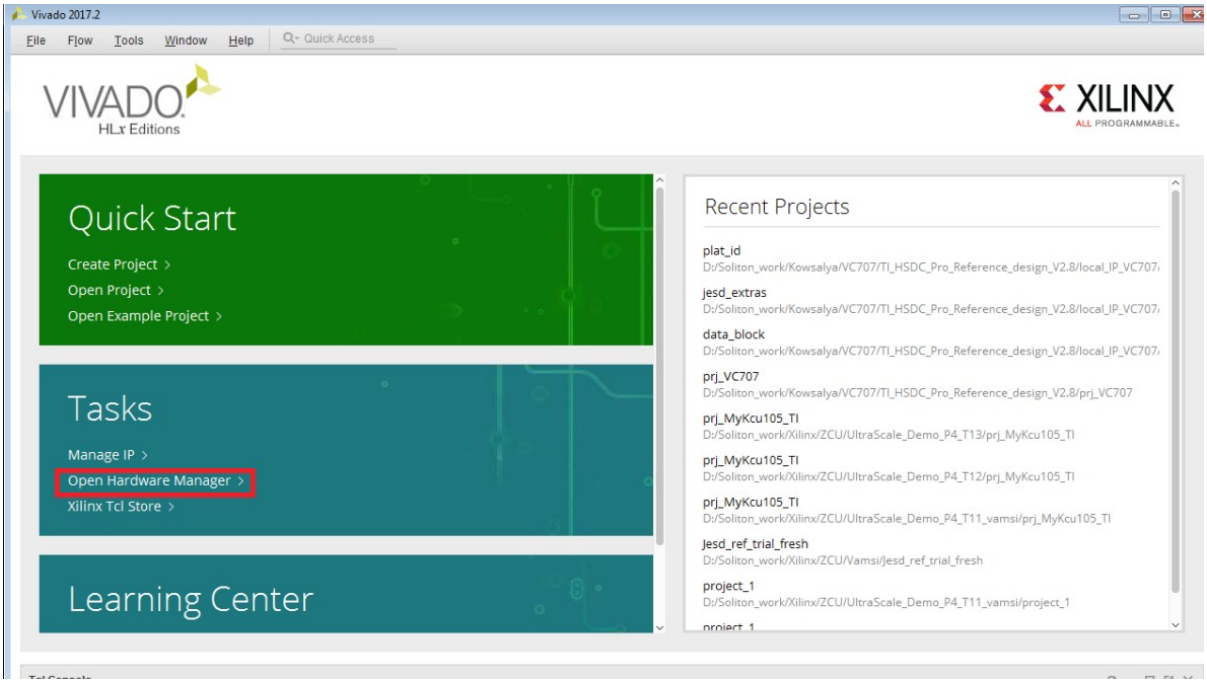
1) Make sure to configure the ADC EVM before programming the device.

2) Open Device manager and check for the COM ports which are named like Silicon Labs Quad CP210x. (Make sure you have connected to the UART port of the board)

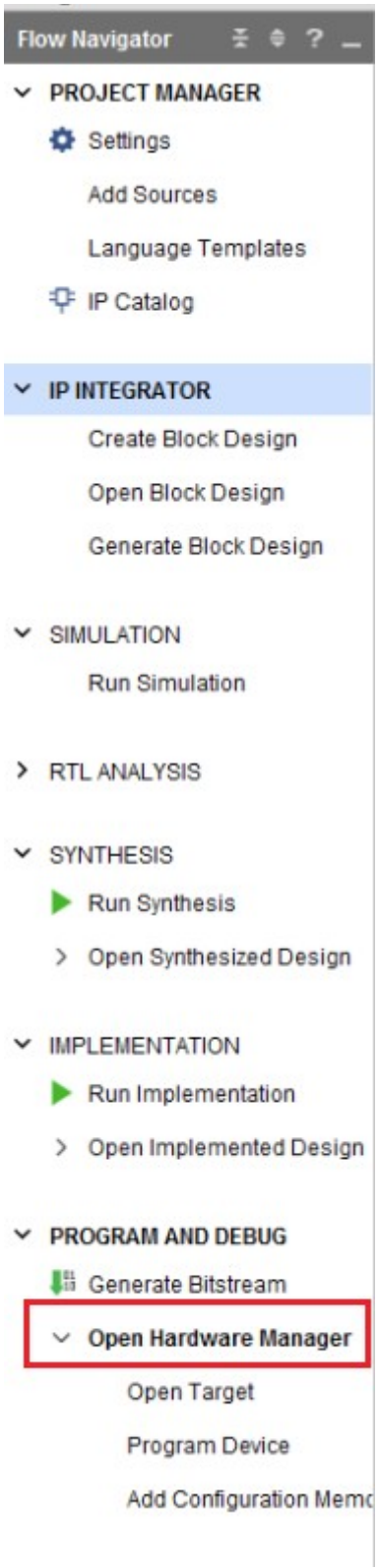


In this case, we have COM Port 3,4,5,6

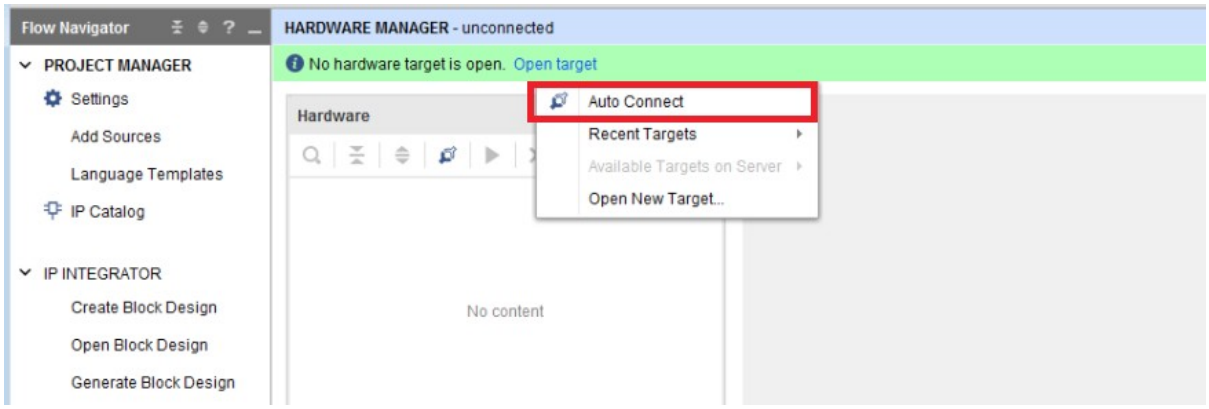
3) Open the Vivado Hardware manager. It can be opened from the GUI start screen (or) the Vivado flow navigator under “Program and Debug”.



(OR)

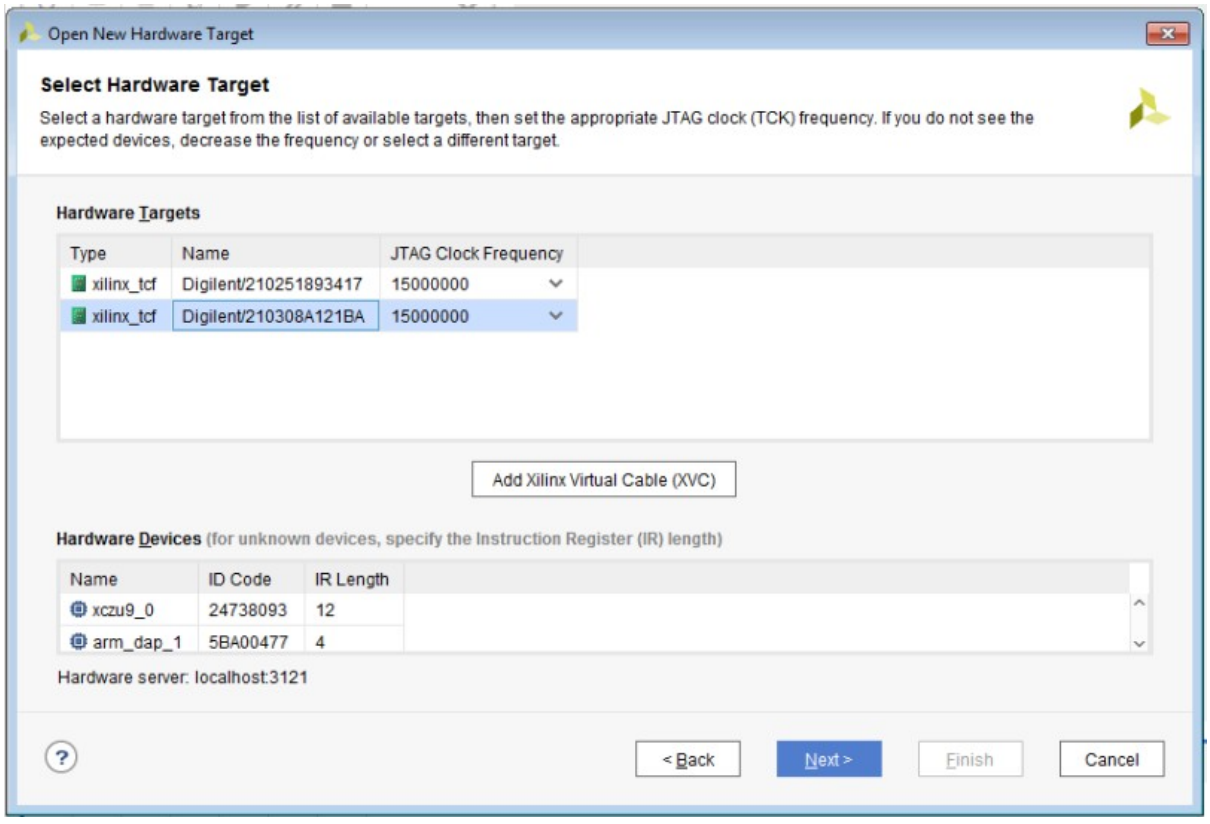

Flow Navigator
PROJECT MANAGER
Settings
Add Sources
Language Templates
IP Catalog
IP INTEGRATOR
Create Block Design
Open Block Design
Generate Block Design
SIMULATION
Run Simulation
RTL ANALYSIS
SYNTHESIS
Run Synthesis
Open Synthesized Design
IMPLEMENTATION
Run Implementation
Open Implemented Design
PROGRAM AND DEBUG
Generate Bitstream
Open Hardware Manager
Open Target
Program Device
Add Configuration Memo

5) Press on Open Target and then press on Auto connect.

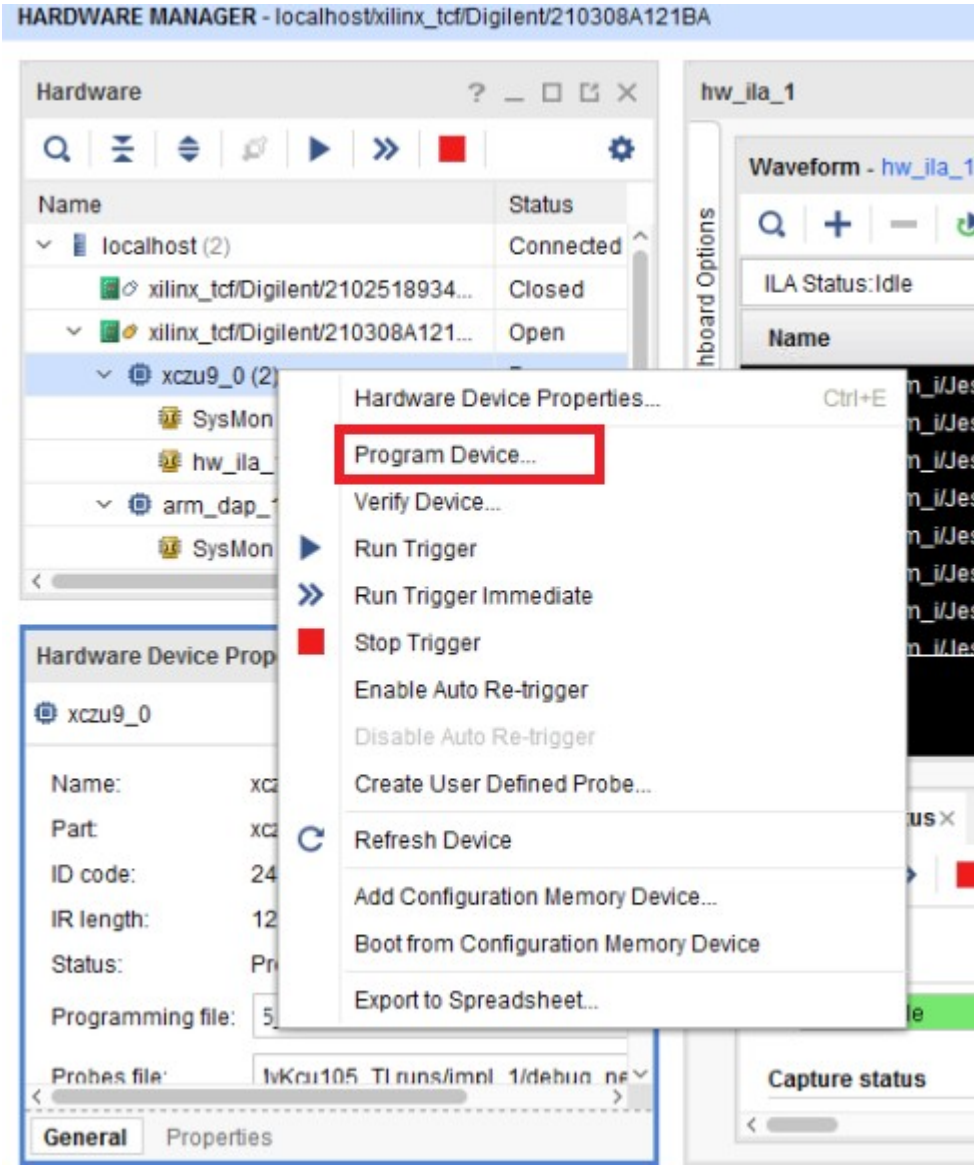


(OR)

If you cannot auto connect, then press on "Open New Target" from the drop down list. Click Next twice and you should see the following.



6) After that Right click on the ZCU102 device part number that is listed and press on Program Device.

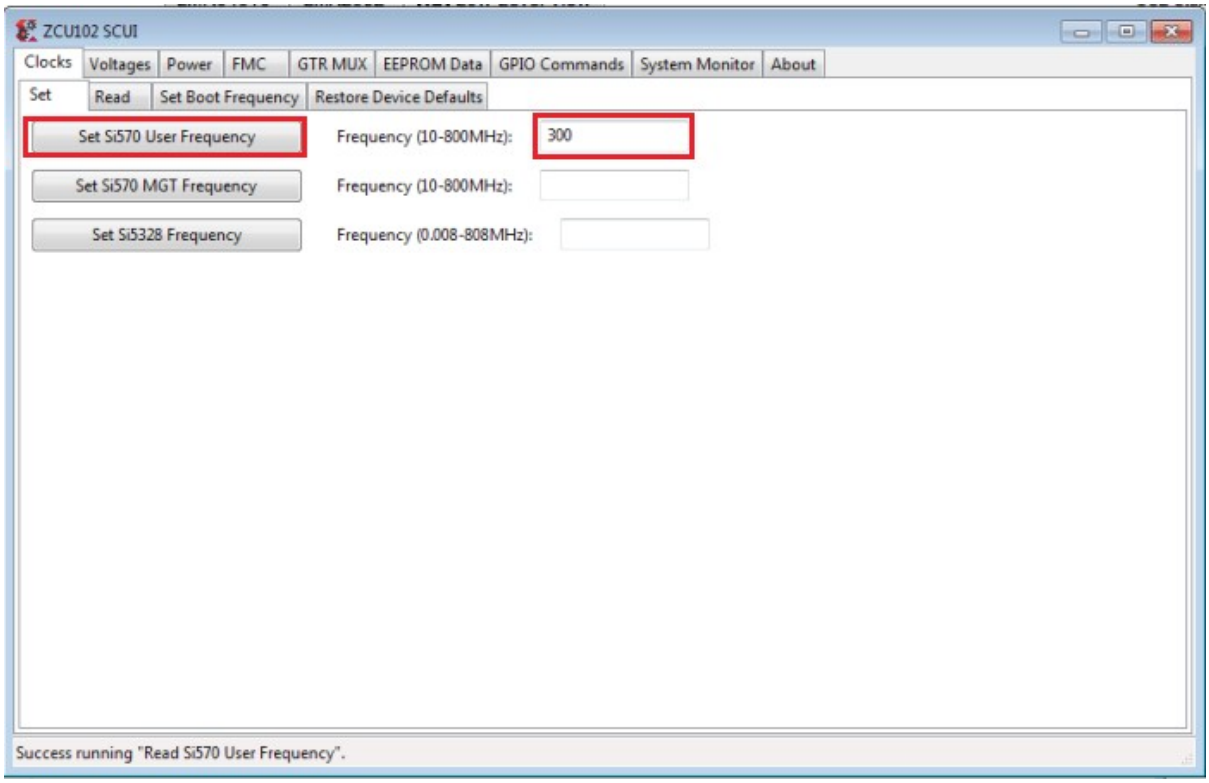


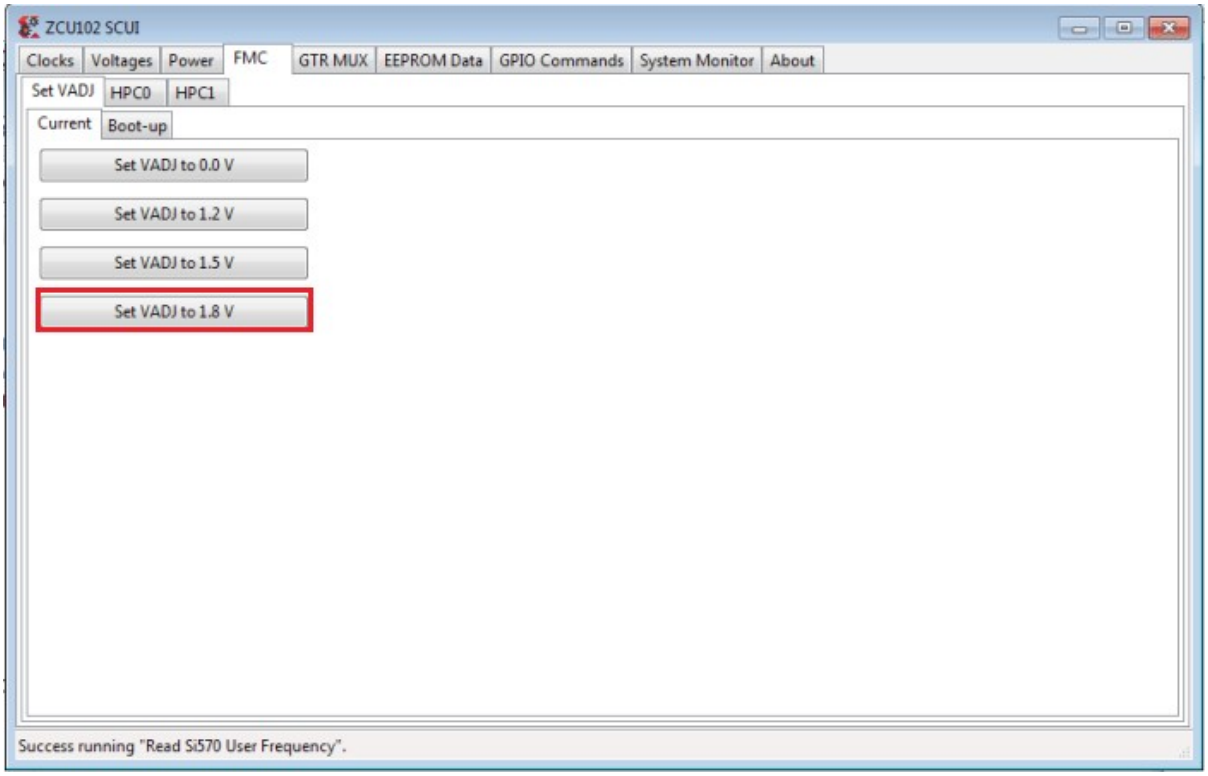
In the Pop-up. Select the "**mySystem_wrapper.bit**" generated earlier in the Bitstream file path. In the Debug Probes file path mention the "**debug_nets.ltx**" file which was also previously generated. Then press "**Program**".

3. **Setting the FMC voltage & User Clock**

We need to set the FMC voltage of the ZCU to 1.8V before we capture. Also the user programmable Si570 clock needs to be set to 300Mhz. In order to set this please make sure that you have connected the UART of the board to the PC. This setting needs to be done only once after the ZCU has been connected and turned ON. For subsequent captures, this section can be skipped.

- Once firmware is downloaded, we need to set the FMC voltage to 1.8v in the enhanced port.
- To do that, Open the System Controller UI from the folder zcu102_scui, by clicking on “SCUI.exe”
- Under FMC -> Se VADJ -> Current, Press on “Set VADJ to 1.8V”
- Then, under Clocks -> Set, Enter the frequency of the Si570 User frequency to 300 and press “Set Si570 User Frequency”.





4. **Checking Results in Chipscope**

Once the development kit is programmed, user can view the results in Chipscope (Vivado Hardware Manager) which probes signals from the board.

In the "**hw_ila_1**" window on the right, you can see number of signals. The different signals have been explained below.

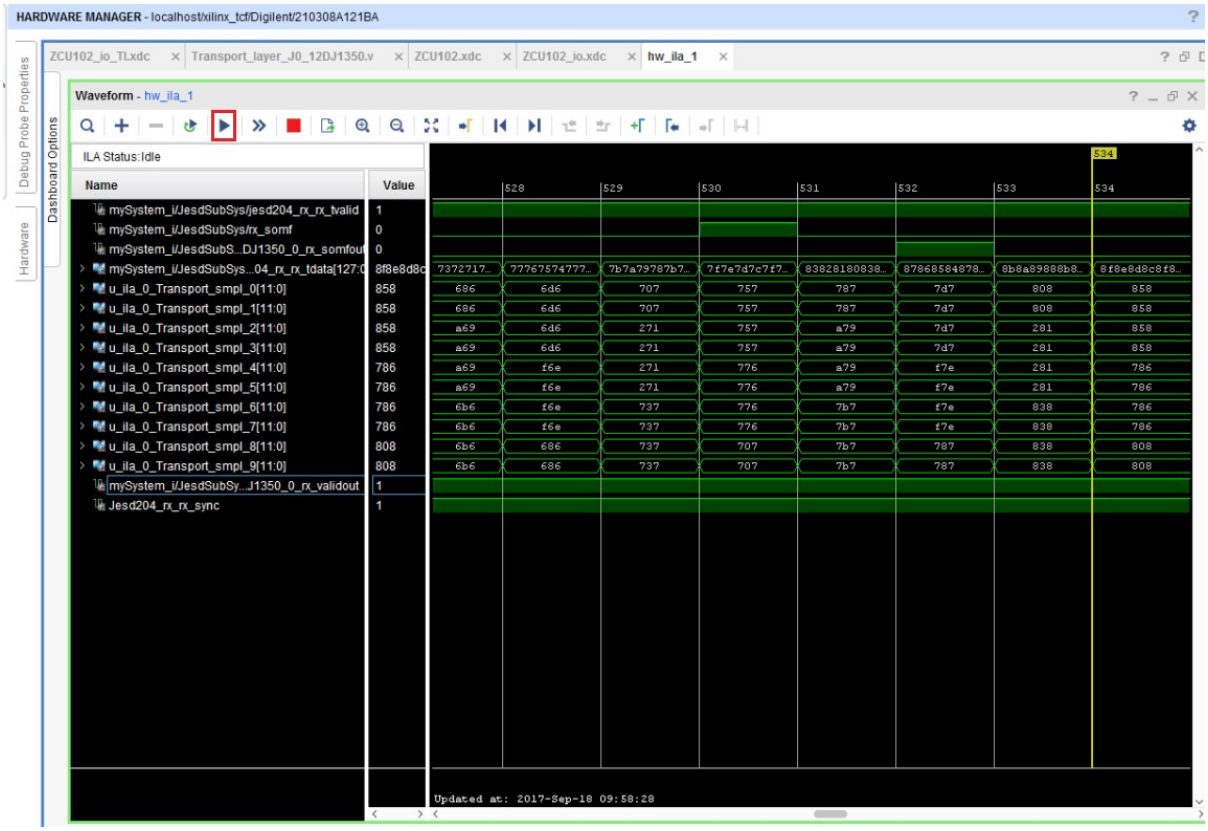
Signals which are currently probed are

- **Rx_sync -** Active low SYNC signal from JESD Base IP. If SYNC is established, this signal will be high.
- **Jesd204_rx_rx_tdata[0…127]** – Data coming from the JESD BASE IP.
- **Jesd204_rx_rx_tvalid** – Indicates whether the output data coming from JESD Base IP is valid. If this signal is low then it is invalid.
- **Rx_somf** – Start of multiframe signal coming from the base.
- **Transport_smpl_0… Transport_smpl_9 –** The output samples coming from the transport layer. The **Jesd204_rx_rx_tdata[0..127]** passes through the transport layer and is converted to a 120 signal bus (Transport_layer_12DJ1350_0_rx_dataout) which contains 10 samples(12bit) per clock cycle. The "Transport_layer_12DJ1350_0_rx_dataout" signal is then sliced into 10 12 bit buses that are probed (Transport_smpl_0… Transport_smpl_9). This has been done for better visibility in the chipscope for the user.
- **Transport_layer_J0_12DJ1350_0_rx_validout -** Indicates whether the output data coming from transport layer is valid. If this signal is low then it is invalid.
- **Rx_somfout** – Delayed SOMF signal coming as an output from transport layer.

Other signals can also be probed. Each time, signals are added/removed from chipscope, the project has to be compiled again.

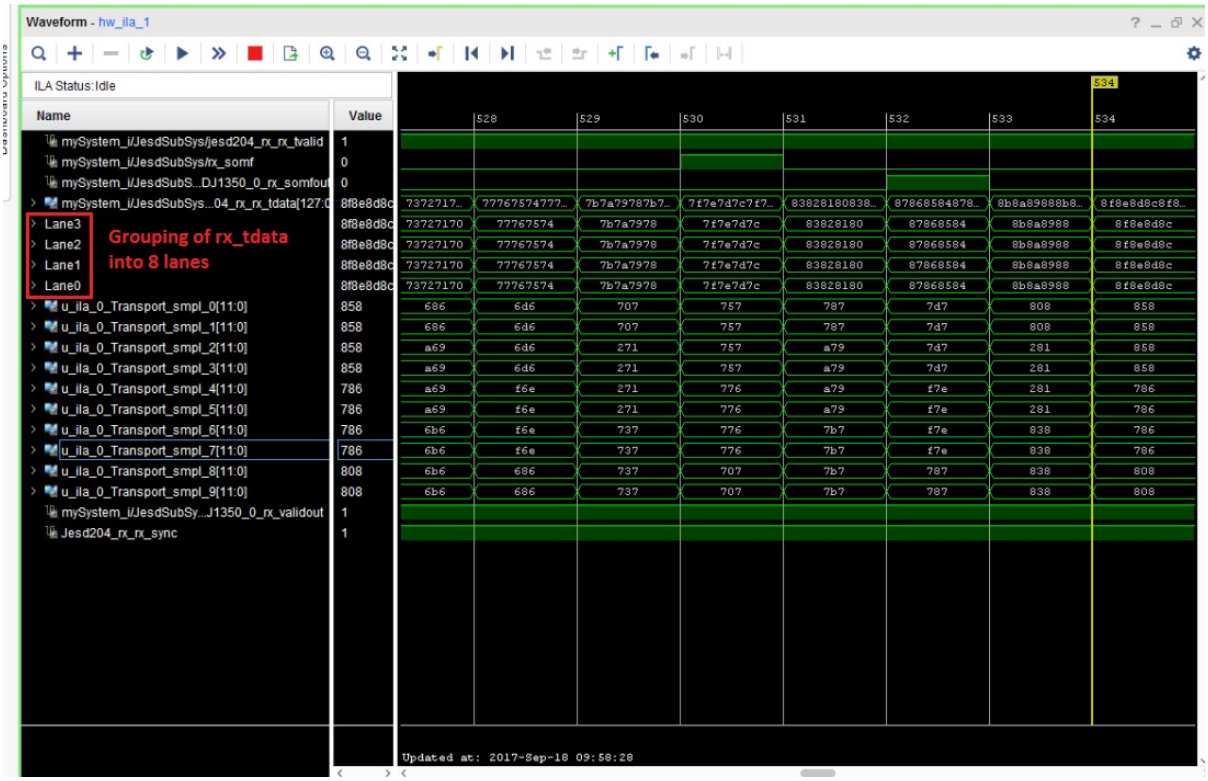
The hw_ila_1 window is shown below.

Press the Play button on the top to capture the data. (data captured in the above image is with Ramp test mode).

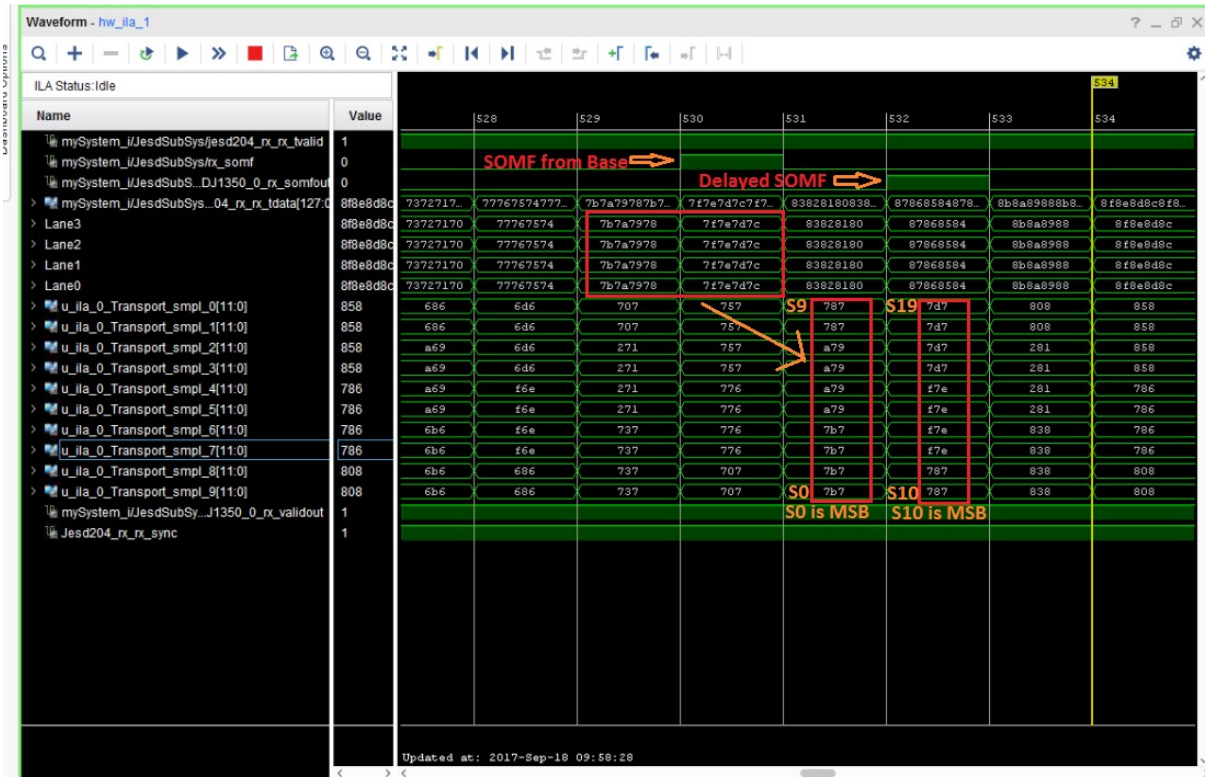


The **Jesd204_rx_rx_tdata** can be grouped into 4 lanes of 32 bits each. This will make viewing much easier . This can be done by selecting the required signals (select 0 to 31 for lane0, 32 to 63 for lane_1 etc.,) and right-clicking and selecting "**New Virtual bus**". Then name the new bus. After grouping into 4 lanes you will get something similar to the image below.

NOTE: The transport layer signals have been sliced in the firmware design itself, so that the user need not group the signals in the chipscope. This part of the block diagram can be removed if the user deems it unnecessary.



Transport layer data out for JMODE0, 8G Lane Rate with Ramp test pattern is below



The transport layer is implemented only for the mode JMODE0.

From the ADC datasheet, the sample format can be obtained as follows

Sample Pattern for one frame of JMODE0

Octet	0		1		2		3		4		5		6		7	
Nibble	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
TX0	A0			A2			A4			A6			A8			T
TX1	A1			A3			A5			A7			A9			T
TX2	B0			B2			B4			B6			B8			T
TX3	B1			B3			B5			B7			B9			T

User should take the transport layer data along with SOMF (Start of Multi-frame) "rx_somfout" signal (Transport_layer_J0_12DJ1350_0_rx_somfout) and rx_validout (Transport_layer_J0_12DJ1350_0_rx_validout) signals. These signals are in alignment with transport layer rx_dataout signal (Transport_layer_12DJ1350_0_rx_dataout).

rx_dataout follows the following sequence for JMODE0

- The first link clock on the rising edge of rx_somfout, rx_dataout signal contains samples from first half of the frame, both channels included

  [A0, B0, A1, B1, A2, B2.., A4, B4] with A0 as MSB

- On the second link clock, rx_dataout contains samples from second half of the frame, both channels included

  [A5, B5, A6, B6, …, A9, B9] with A5 as MSB

- On the third link clock, rx_dataout contains samples from the first half of successive frame, both channels included

  [A0, B0, A1, B1, A2, B2.., A4, B4] with A0 as MSB and the sequence repeats.

V. **STATUS LEDS**

Two signals have been added in the top module for debugging

rx_sync: This signal refers to the SYNC out from JESD Base IP and is given to LED D0 on board. It will be OFF if SYNC is lost. Under normal process, this LED will be ON

rxoutclk_bf: This signal indicates if the link clock (lane rate/40 clock) generated from the PHY module and passes through a buffer before being connected to the LED. This is connected to LED D4.

Apart from the above two LEDs, few other signals are assigned to LED mainly to prevent logic deletion by Fitter tool and it can be ignored

Note: Both the LEDs are active high