ARRIA 10 AFE74xx XCVR 2x44210 JESD REFERENCE DESIGN USER GUIDE

Table of Contents

I. DESCRIPTION

Overview:

The "A10_AFE74xx_XCVR_2x44210_7p3728G.qar" is a reference design developed to target Arria10 EVM board to interface with AFE74xx through JESD protocol. It is a transceiver firmware and both the Rx and Tx section of firmware supports the JESD mode 2x44210 at lane rate 7.3728G.

Rx design description:

RX side of FW has *altera_jesd204b_rx* IP and *xcvr_jesd_rx* IP to get JESD data from the AFE74xx at lane rate 7.3728G lane rate. The design has a simple Rx transport layer specific to this mode (2x44210) that captures samples from the ADC, re-order the bits and give out 16 samples every link clock (2 samples per lane). The results can be verified using Signal Tap.

Tx design description:

TX side of FW has *altera_jesd204b_tx* IP and *xcvr_jesd_tx* IP to send digital data (Sine wave 10MHz) to the DAC through JESD interface and it operates at lane rate of 7.3728 G. The design has a simple Tx transport layer which generates sine and cosine samples by the DDS compiler in firmware. The generated samples are reordered as per the JESD mode (2x44210), 2 samples per lane, with 4 lane for I data and other four lanes for Q data and sends it to the *xcvr_jesd_tx* IP. The results can be verified using a scope or Spectrum analyzer.

Note: This version of the FW is a fixed line rate firmware and hence will work only at 7.3728G. For any other line rate the firmware needs to be recompiled for that specific line rate.

This document gives a brief on the compilation and verification process involved. Section II discusses on how to restore the project from .qar file and the compilation process involved. Section III discusses on how to get the data in Signal Tap and validate the same. Section IV discusses on generation of tone for DAC. Section V discusses the debug signals added in project.

Following are some of the important inputs and the output signals with their description. User can make use of these signals in their custom modules.

Signals	Description	Direction
device_clk	reference clock from AFE74xx	Input
mgmt_clk	100MHz oscillator clock	
global_rst_n	Active Low User reset	
sysref	SYSREF Signal	
tx_syncn	Tx Sync Signal	
rx_serial_data	Serial Data from ADC	
tx_serial_data	Serial Data to DAC	Output
rx_dataout	Transport Layer data output (16 samples for every link clock)	Output from the Rx transport layer module
rx_somfout	SOMF aligned with rx_dataout	
rx_validout	Data valid Signal aligned with rx_dataout	
Jesd204_tx_link_data	Transport Layer data out (16 samples for every link clock)	Output from the Tx transport layer module

Jesd204_tx_link_valid	Valid signal of the transport layer data	

Note: Tx SYNC signal of the device routed to FMC is not connected to FPGA in the Arria10 EVM. Hence the SYNCB0CMOS of the device is bluewired to Arria10 EVM through J7 SMA connector.

In AFE74xx used for testing, the lower four SERDES lanes P & N pins of the JESD interface are swapped, hence Rx lane polarity inversion is implemented in the design to address that. Due to this, Rx lane polarity inversion constant given to the JESD IP module (csr_rx_lane_polarity) is '0x0F'. But on the Tx side no lane inversion is required hence Tx lane polarity inversion constant given to the JESD IP module (csr_tx_lane_polarity) is '0x00'

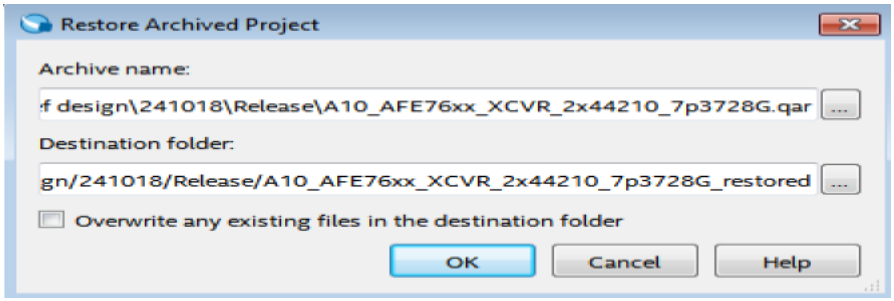
II. COMPILING QUARTUS PROJECT

1. Restore the Quartus project

Open Quartus 16.1. Click File -> Open and choose the qar file

"A10_AFE74xx_XCVR_2x44210_7p3728G.qar"

Click OK in the dialog box that pops up next



Quartus will restore the contents to A10_AFE74xx_XCVR_2x44210_7p3728G_restored folder in the same location as that of qar file. User can change the location or the name of the folder.

2. Compiling the project

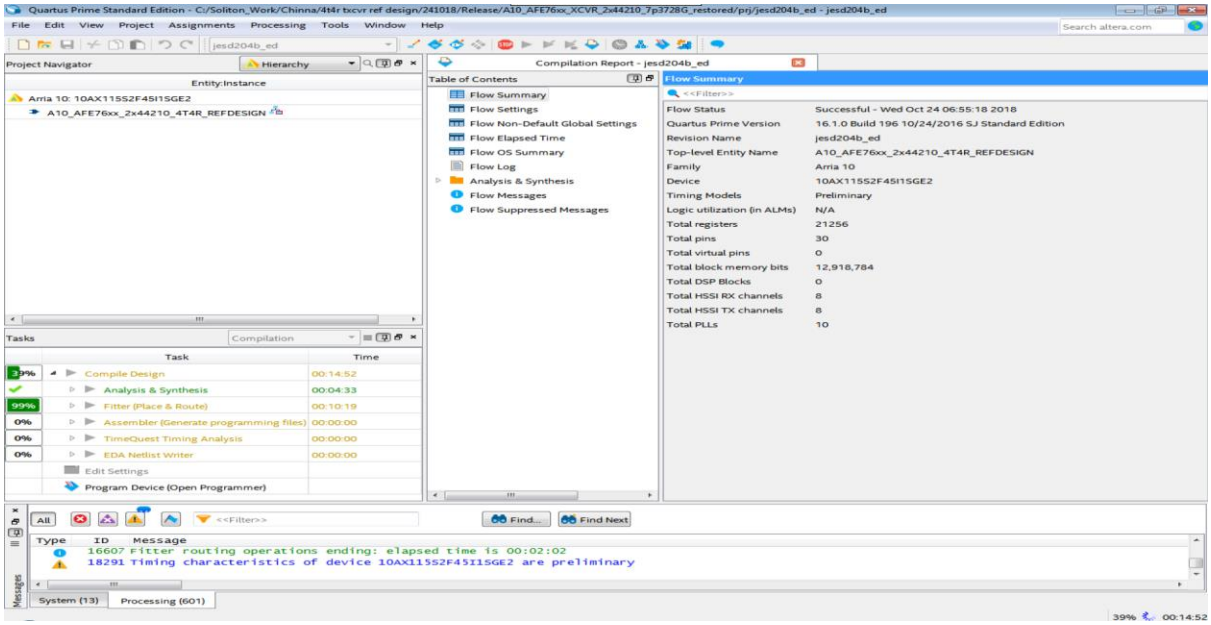
In Quartus IDE (environment), go to Project Navigator window located left side. This will show the device for which the firmware was compiled and the hierarchy of Verilog modules used.

Tasks window right below the Project Navigator will list all the tasks involved in compiling a design. To compile the project, double click "Compile Design" and all the subtasks listed will be executed in sequence

This will generate a ".SOF file" in the following location

Relative path of the example design +

"\A10_AFE74xx_XCVR_2x44210_7p3728G _restored\prj\output_files"

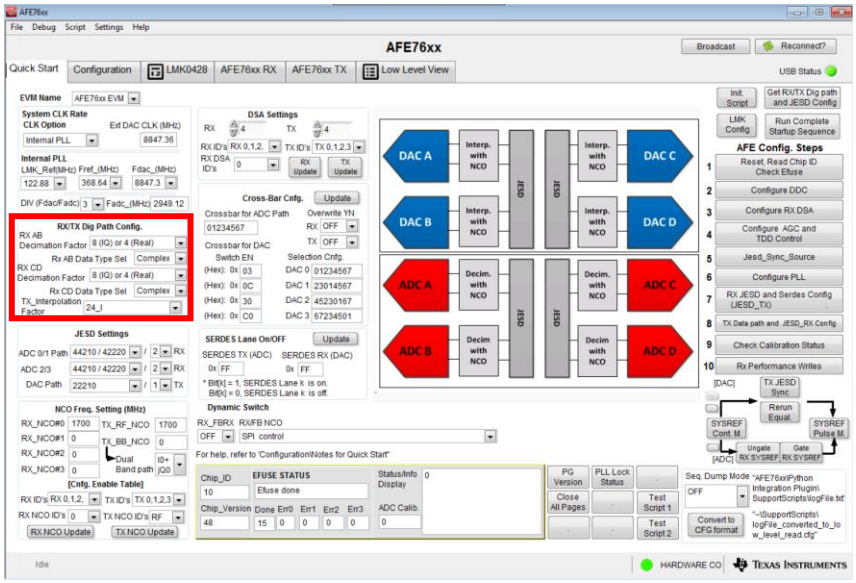


Once the SOF file is generated, proceed to next section. It will take 20 min approximately to compile the design

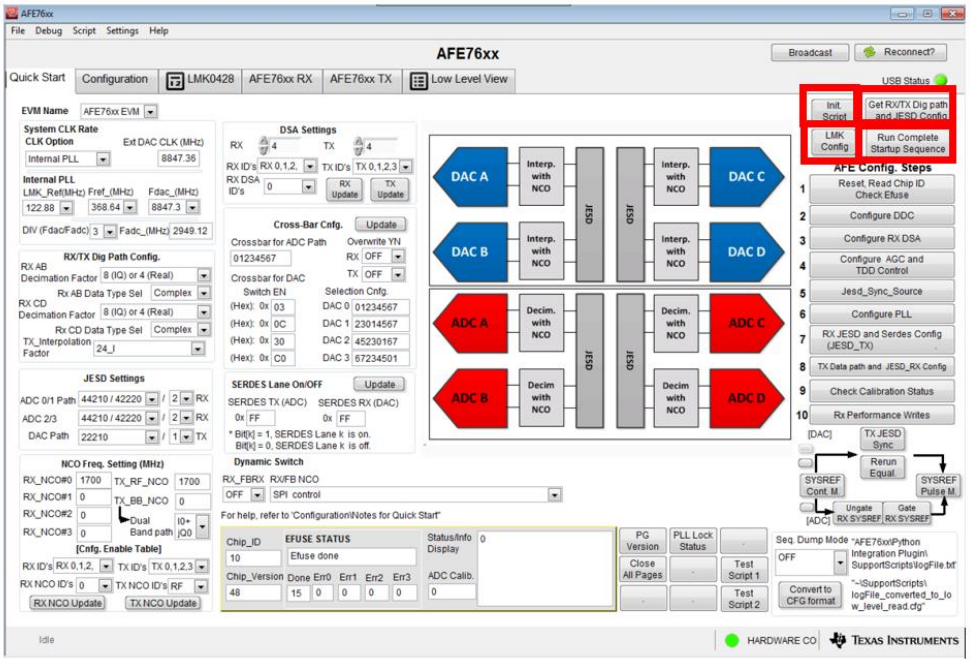
III. **VALIDATING WITH AFE74xx**

1. **Configure ADC**

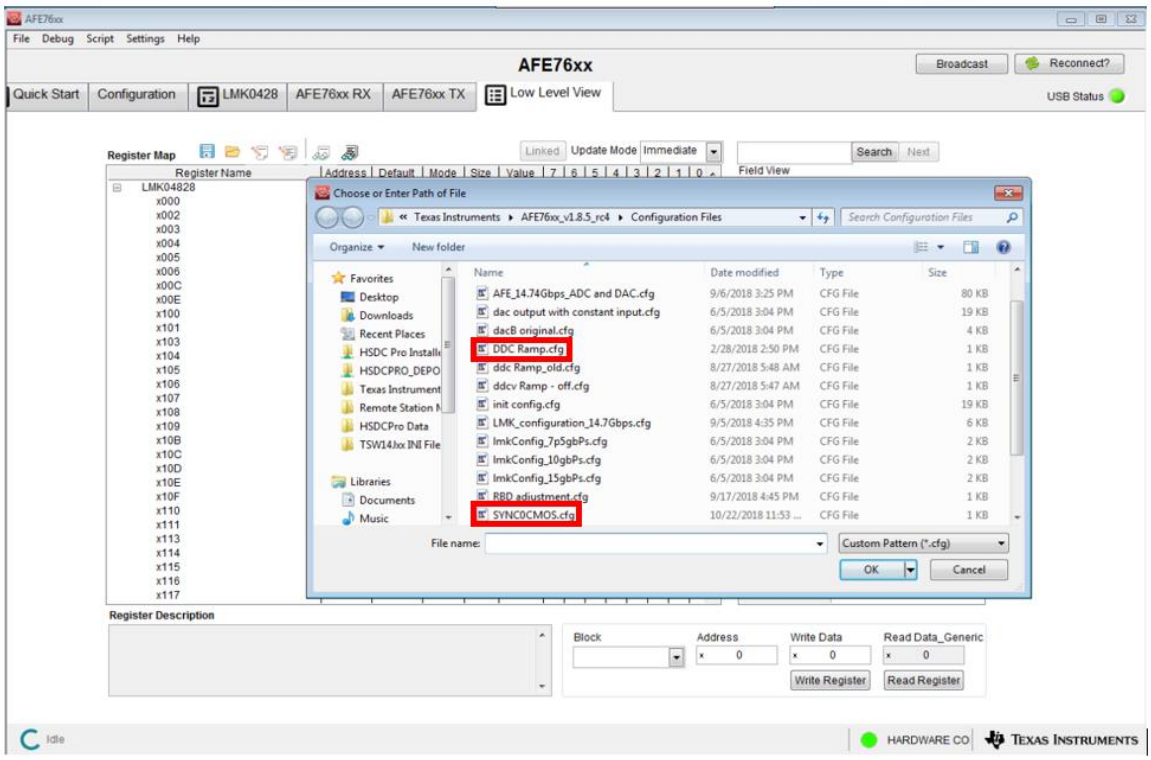
- The reference clock from device should be stable before downloading firmware. So, the device has to be configured first before we program the board.
- Open AFE GUI (Version 1.8.5 RC4) and set values as shown in the below snapshot



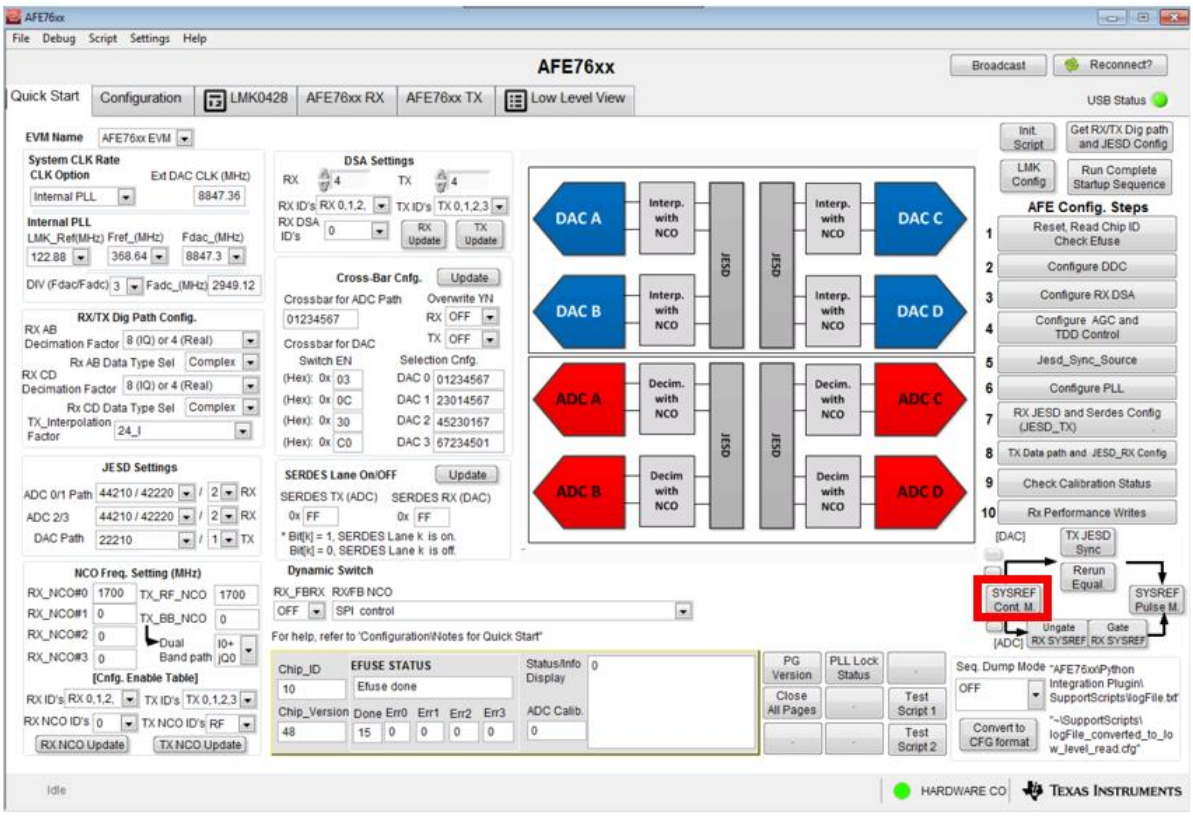
- In AFE GUI press buttons ‘init Script’, ‘Get Rx/Tx Dig path and JESD Config’, ‘LMK Config’ & ‘Run Complete Startup Sequence’ in sequence



- Press on the small folder icon in the ‘AFE GUI’ and load ‘SYNC0CMOS’ and ‘DDC Ramp’ config files

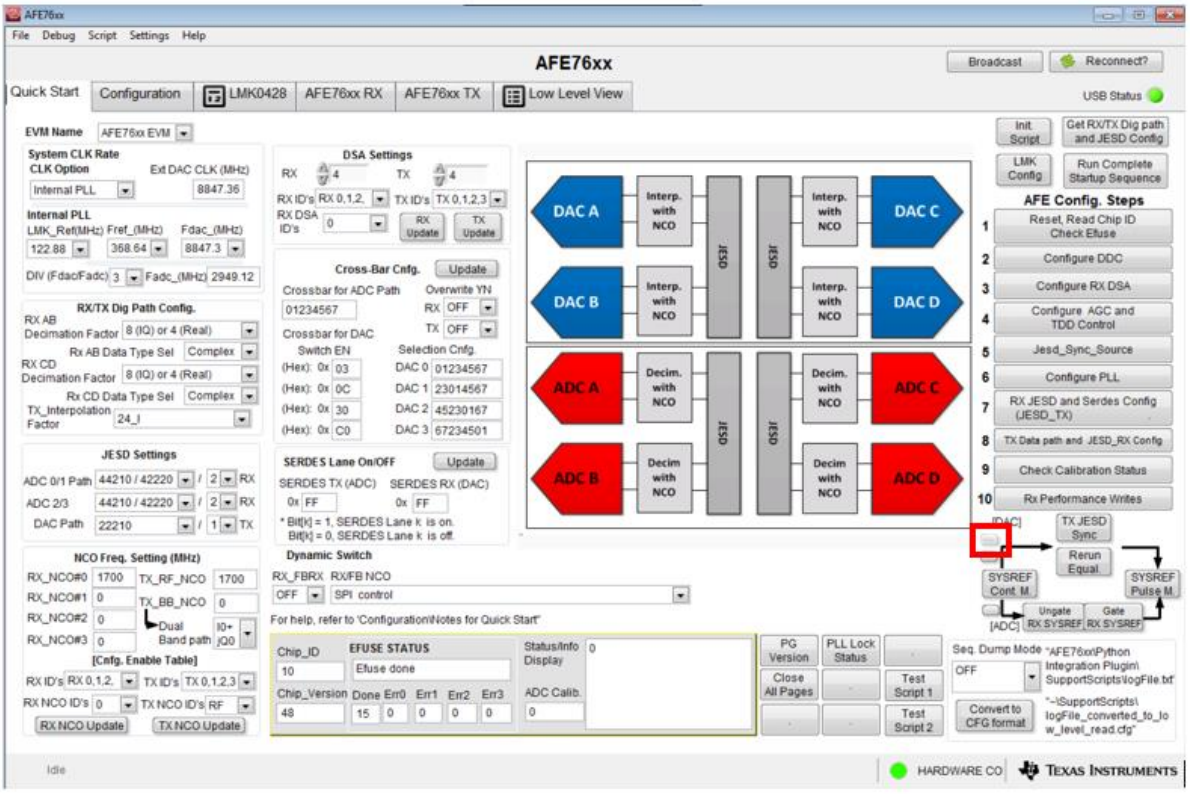


- Press the ‘SYSREF Cont M’ button in AFE GUIs



- Download Firmware (Refer Section 2)

- Press on DAC SYNC button (highlighted in the snapshot below) in AFE GUI

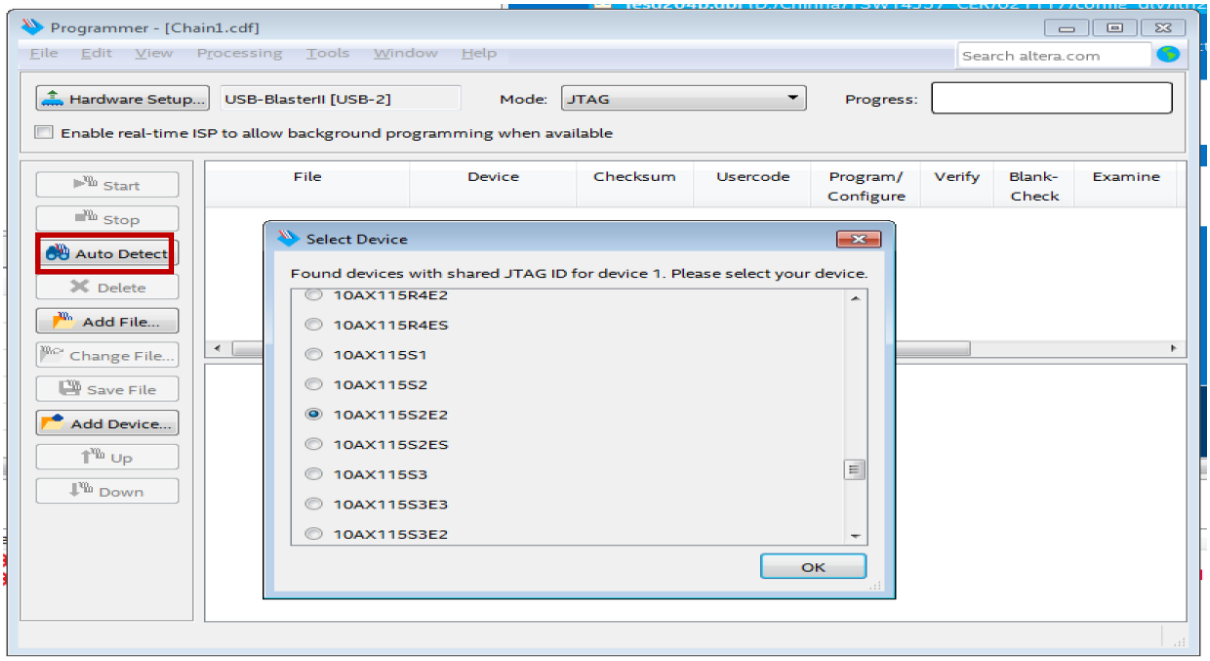


With this step we should be able to see Ramp data from ADC and a clear tone from DAC

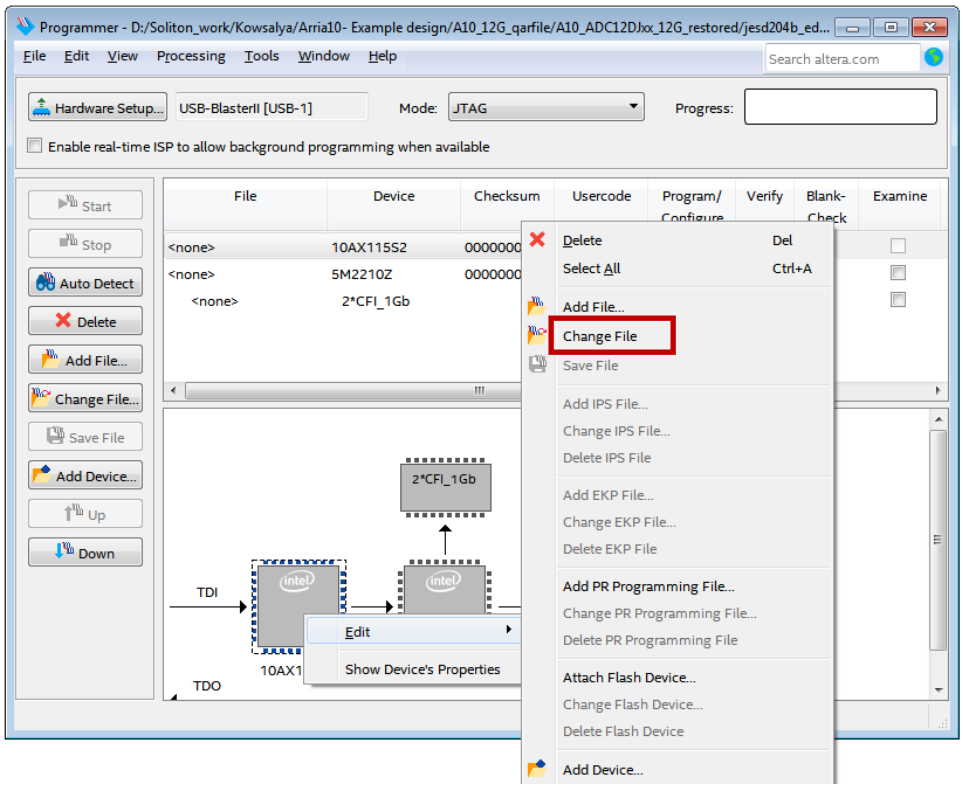
2. Programming Arria 10 development kit

Connect USB Blaster to Arria10 development kit. Go back to Quartus project, and double click Program Device in tasks window. This will open the programmer tool.

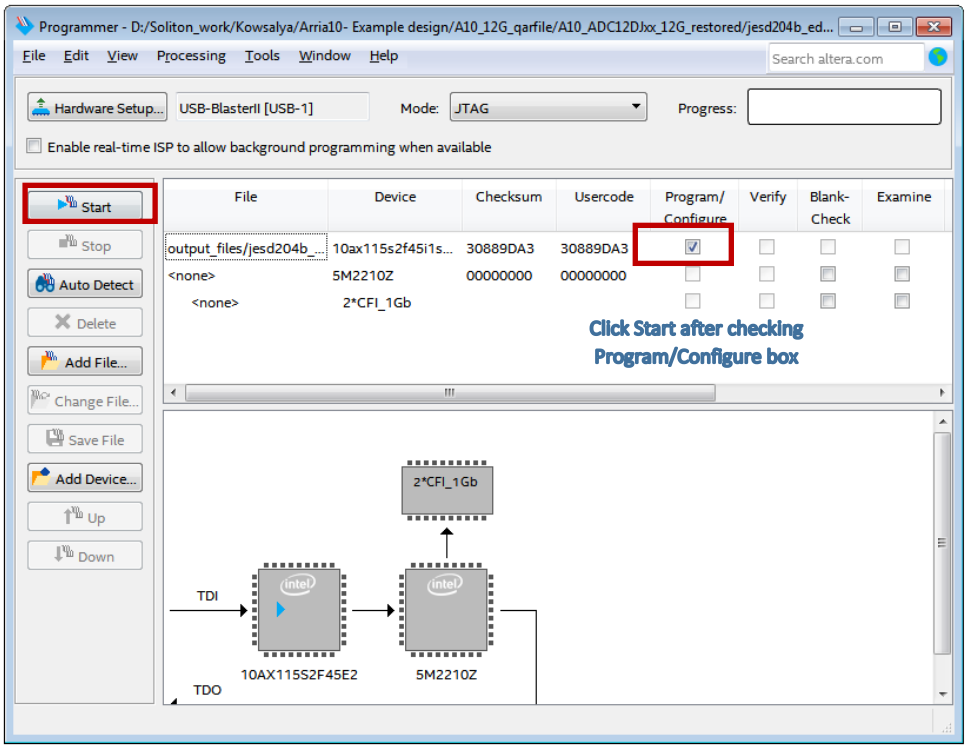
Click on Auto Detect and choose “10AX115S2E2” in the Select Device window and click OK



Right click on “10AX115S2E2” in the JTAG chain and choose Edit-> Change File. Browse and select the .SOF file generated in previous section

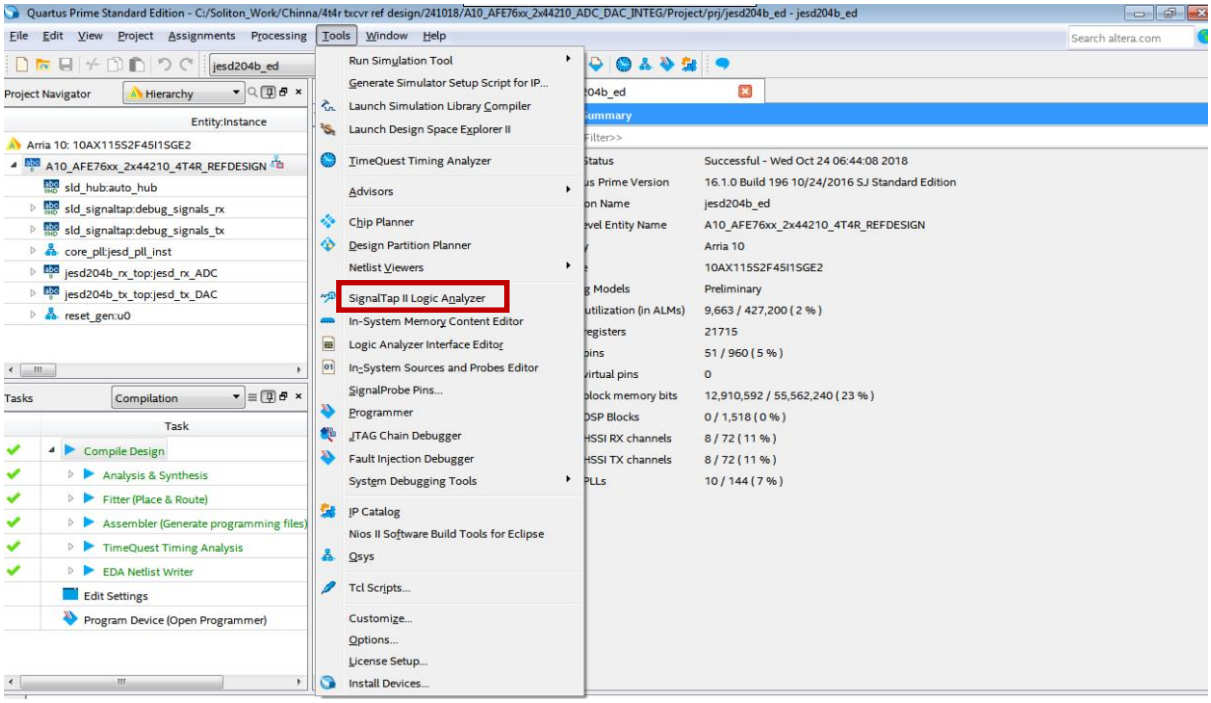


Check the Program/Configure box and click start. Progress bar should start incrementing



3. Checking Results in Signal Tap

Once the development kit is programmed, user can view the results in Signal Tap which probes signals from the kit using USB blaster and JTAG interface. It can be opened from Quartus Tools-> SignalTap II Logic Analyzer



Signals which are currently probed are

Signals in Rx Instance:

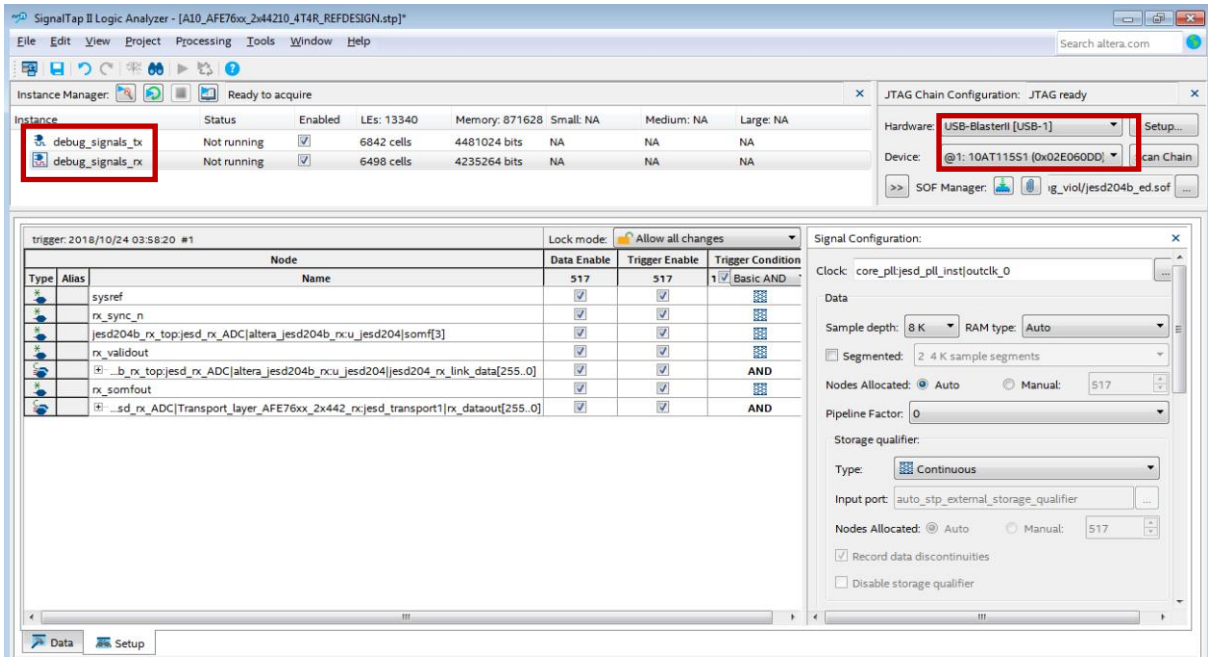
- **rx_sync_n** – Active low SYNC signal from *altera_jesd204b_rx* IP. If SYNC is established, this signal will be high
- **sysref** – SYSREF signal to Base IP
- **somf[3]**- SOMF Signal from *altera_jesd204b_rx* IP
- **Jesd204_rx_link_data** – 256 bit link data from *altera_jesd204b_rx* IP (grouped into 8 lanes of 32 bit each)
- **rx_dataout**- transport layer data out (16 samples of data for every link clock). User has to use rx_dataout along with rx_somfout and rx_validout
- **rx_somfout**- SOMF signal aligned with transport layer rx_dataout
- **rx_validout**- Data valid signal aligned with transport layer rx_dataout

Signals in Tx Instance:

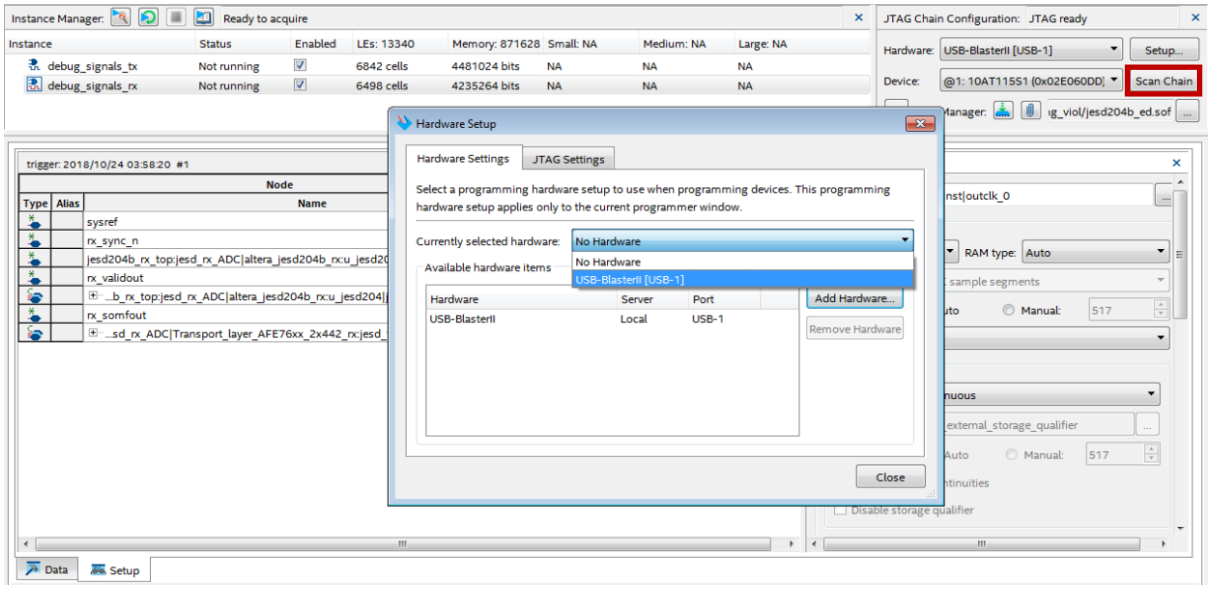
- **tx_sync_n** – Active low SYNC signal to *altera_jesd204b_tx* IP. If SYNC is established, this signal will be high
- **sysref** – SYSREF signal to Base IP
- **jesd204_tx_pcs_data** – 256 bit pcs data from *altera_jesd204b_tx* IP (grouped to 8 lanes with 32 bit data for each lane)
- **ddsio|fsin_o & ddsio|fsin_1 –** The 16 bit output samples from each DDS compiler (see DAC wave form generation section).
- **jesd204_tx_link_valid** - Data valid signal aligned with transport layer jesd204_tx_link_data.

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

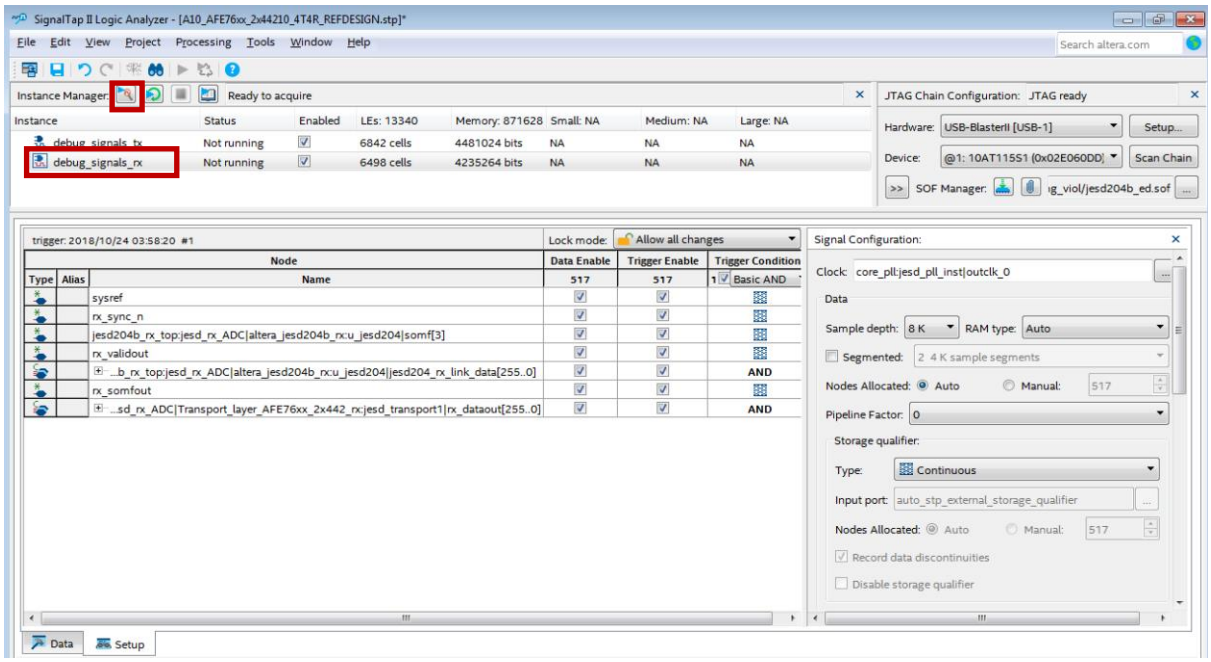
Following is the Signal Tap window. Hardware and Device tabs should be selected/listed properly. The Instance section lists both the Tx and Rx instances. Double clicking on any particular instance will display its corresponding signals.



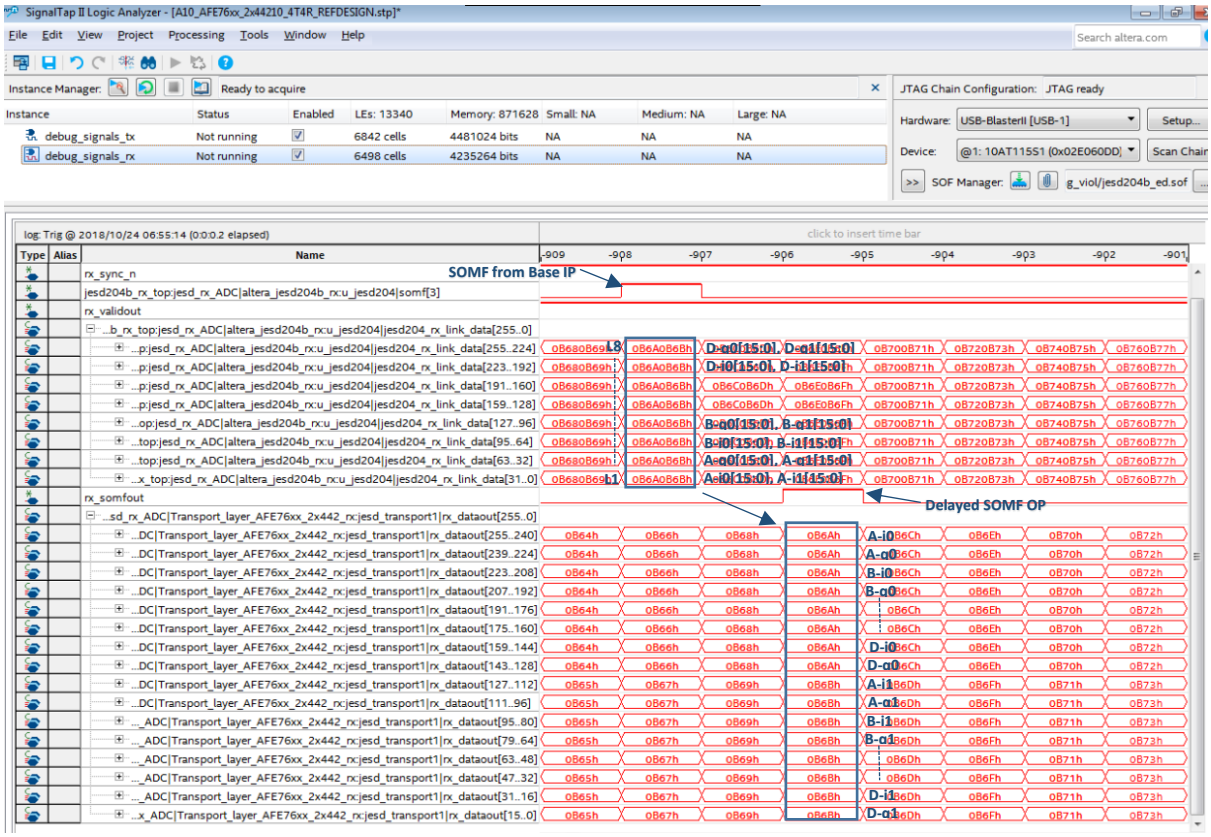
If Hardware or device is not listed, click on Setup and choose the right port to which USB blaster is connected



Select debug_signal_rx and Click on Run Analysis



Transport layer data out for 2x44210, 7.3728G Lane Rate with Ramp test pattern is below.



The transport layer is implemented only for the mode 2x2RX_44210.

From the ADC datasheet, the sample format of JESD mode 2x44210 is as follows

Octet	1	2	3	4
Lane 0	A-i0		A-i1	
Lane 1	A-q0		A-q1	
Lane 2	B-i0		B-i1	
Lane 3	B-q0		B-q1	
Lane 4	C-i0		C-i1	
Lane 5	C-q0		C-q1	
Lane 6	D-i0		D-i1	
Lane 7	D-q0		D-q11	

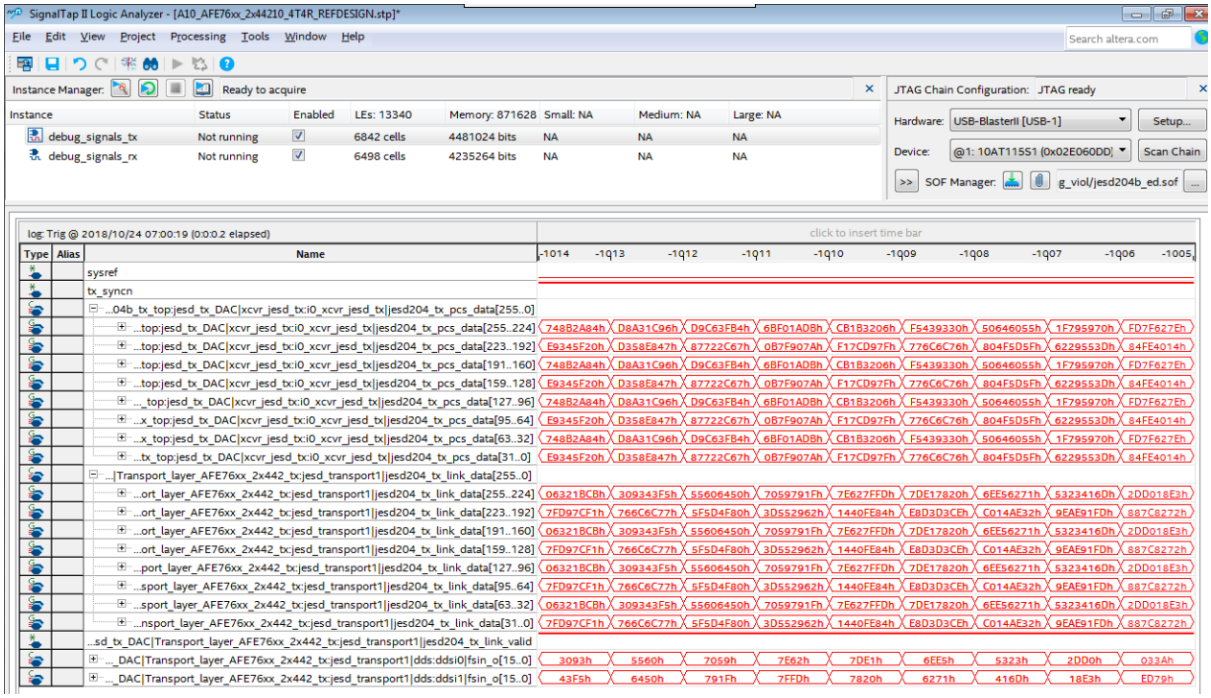
Note: The samples specified as A & B are from one device and samples C & D from the other.

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

rx_dataout follows the following sequence for 2x2Rx_44210

- The first link clock on the rising edge of rx_somfout, rx_dataout signal contains two 16 bit samples for each channel, with samples as [A-i0, A-q0,…,D-q0] on the MSB 128 bits with A-i0 as MSB and [A-i1,A-q1,…,D-q11] on the LSB 128 bits with A-i1 in MSB.

Similarly select the TX instance 'debug_signal_tx' and Click on Run Analysis



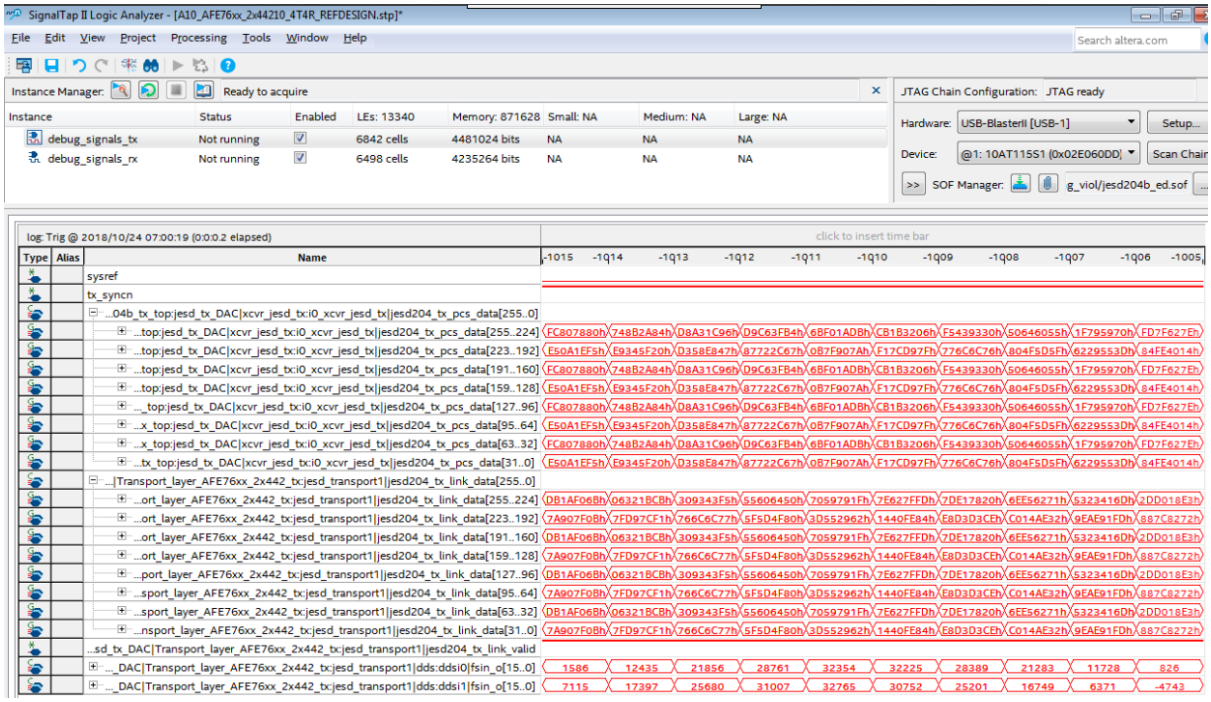
- 'jesd204_tx_pcs_data' shows the 256 bits of data which will be sent to DAC serially,
  It is grouped as 32 bit buses so that each bus corresponds to data of a single lane.
- 'jesd204_tx_link_data' shows the 256 bits of data which is formed by rearranging the data generated by the DDS module so that it suits 2x4421 mode of DAC.
  It is grouped as 32 bit buses so that each bus corresponds to data of a single lane.

- ‘ddsi*|fsin_0’ shows the 16 bits of sample data (sine wave of 10 MHz) which is generated by the DDS module.

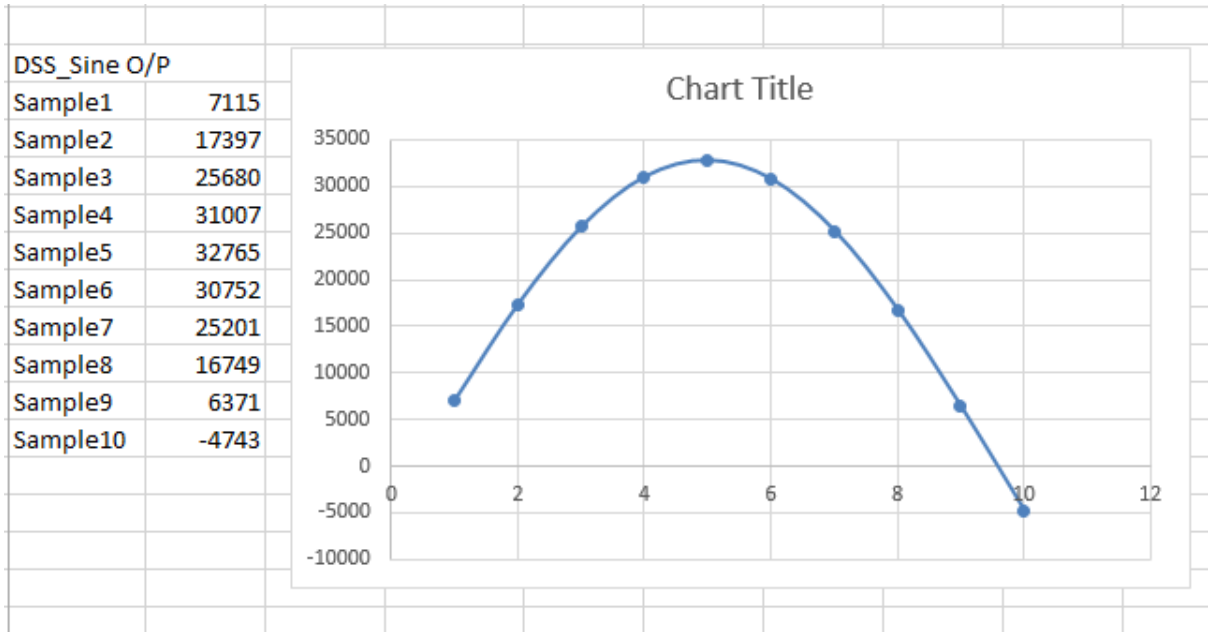
It is possible to view the output of each DDS as a 10 MHz wave, within the chipscope.

The procedure:

1. Right click -> Bus Display Format -> Signed decimal in Two’s Complement

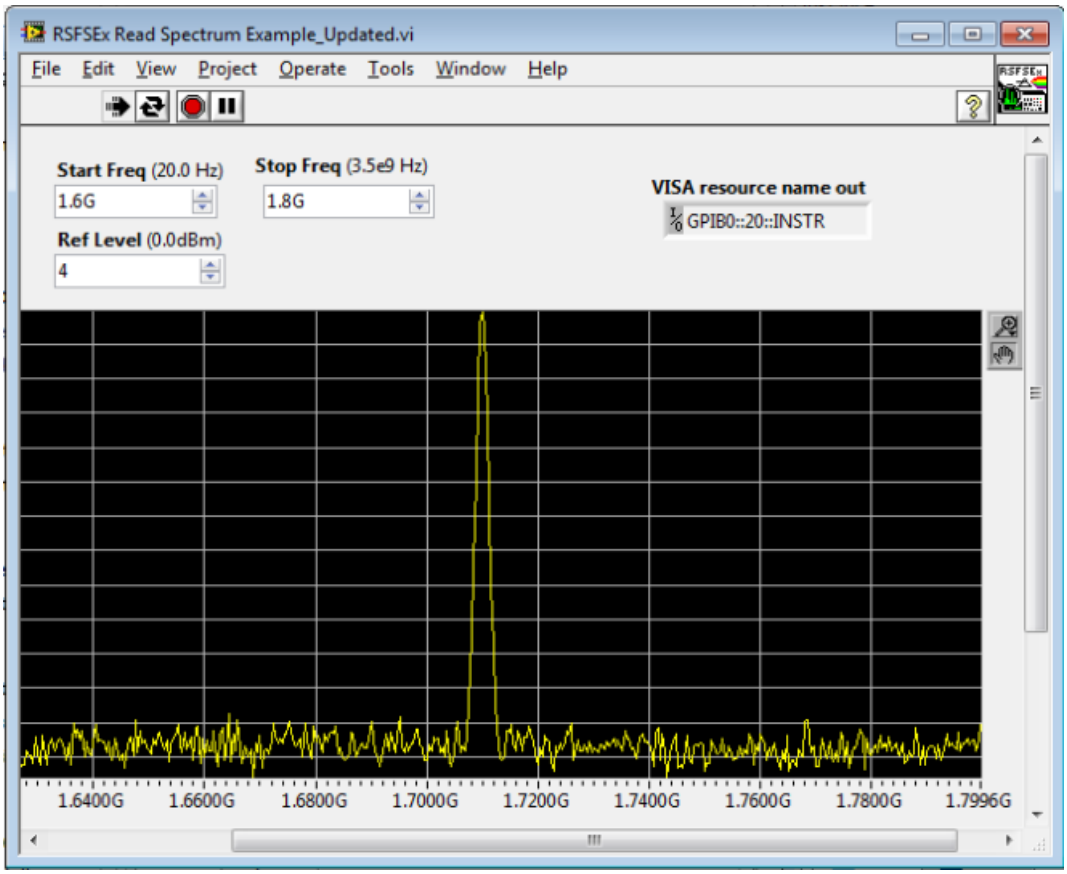


- We can plot the samples of any dds instance ouput(fsin_o) and we should get a sine wave as follows(plot of ‘ddsi0|fsin_o[15:0]’)



4. Spectrum Analyzer

The frequency as seen in the Spectrum Analyser:



Note: As NCO frequency is set to 1.7G and the generated frequency is 10 MHz, we see a tone from the DAC at NCO Freq+ Generated frequency i.e. 1.71G

IV. DAC wave form generation

This FW (and Transport Layer) has been made specifically for 2x4421 mode. A tone is generated within the FW. This tone is being sent continuously (free-running) to the DAC from the FPGA.

1. Generation of Sinewave:

- A DDS Compiler is being used within the FW in order to generate the tone. In this design, a Sine wave of frequency of 10 MHz is being generated.
- The DDS compiler will require a sampling clock as input. In this firmware, the sampling frequency of the DDS compiler is 184.32 MHz
- For every link clock cycle we need to send 32 bits of data to the *altera_jesd204b_tx* IP. Hence we use 2 instances of DDS compiler modules to generate 2 waves of 10 MHz each. However

each instance is offset by an equal value (for this mode the phase offset is one-half of the output wave i.e., 10/2 = 5Mhz) such that in each link clock 2 samples of a 10 MHz Stone is generated. All two samples are concatenated (jesd204_tx_link_data signal) and given as an input to the *altera_jesd204b_tx* IP.

- The concatenated output (jesd204_tx_link_data) consists of four 16-bit samples. In this mode 2 samples are provided for each lane. Hence the output of the transport layer is a 256 bit bus, which contains sixteen 16-bit samples for 8 lanes.
- The DAC sampling rate for this lane rate (7.372G) is 368.64 Msps/ Channel. Each DDS Compiler instance generates a 10 MHz wave w.r.t a sampling rate of 184.32 MHz Therefore when we combine the outputs of two such instances for a single cycle, a tone of 10 Mhz is generated for a sampling rate of 368 Msps.

2. **Changing the Sine wave frequency**:

- In order to change the generated Sine wave frequency, we need to change the phase offset and phase increment values.

  (Calculation has been presented for 20 MHz sine wave case)

  - Phase offset = [2^Phase width /(Sampling frequency/Required frequency)]/2
    = [2^32/(184.32/10)]/2
    = [4294967296/18.432]/2
    = 233016888.889/2
    = 116508444.444

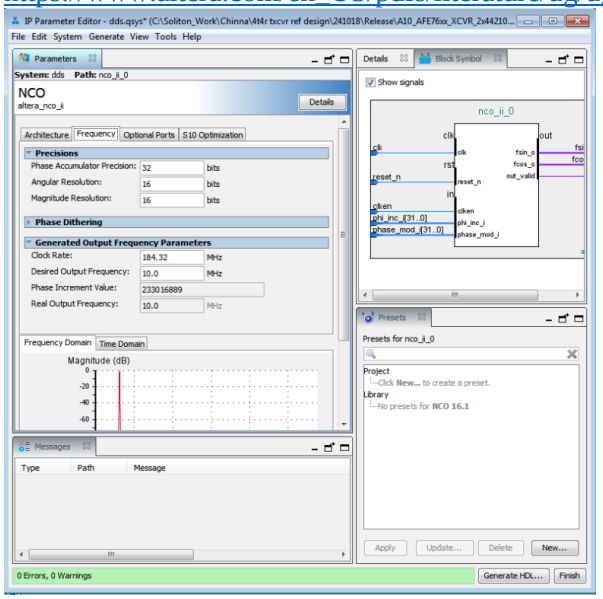
    By rounding off to the nearest integer we get 116508444 which is 6F1FC71C in hexadecimal.

  - Phase increment = 2^Phase width /(Sampling frequency/Required frequency)
    = 2^32/(184.32/10)
    = 4294967296/18.432
    = 233016888.889

    By rounding off to the nearest integer and making it a multiple of phase offset, we get 23301689 which is DE38E38 in hexadecimal.

For more detailed information about the DDS compiler module please refer to the Altera document available at the following link:

https://www.altera.com/en_US/pdfs/literature/ug/ug_nco.pdf



V. **STATUS LEDS**

Few signals added for debugging are listed below,

link_clk_led: This signal indicates if the link clock (lane rate/40 clock) from ADC is available or not. A signal which toggles for every 160ms, derived from link clock is connected to this LED(D7).

rx_altsyncn_led: This signal refers to the SYNC out from *altera_jesd204b_rx* IP and is given to LED D6 on board. It will be ON if SYNC is lost. Under normal process, this LED will be OFF.
tx_syncn_led: This signal refers to the SYNC input to *altera_jesd204b_tx* IP and is given to LED D9 on board. It will be ON if SYNC is lost. Under normal process, this LED will be OFF

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 low