ARRIA 10 DAC38RF82 JESD REFERENCE DESIGN USER GUIDE

Table of Contents

I. DESCRIPTION

The design “A10_DAC38RF82_7p68G_84111.qar” is developed for Arria10 development kit (reference link) for the mode: 84111 with lane rate 7.68 Gbps. It has JESD Base IP and JESD PHY IP to send DAC data (Sine wave 20MHz) to the DAC through JESD interface. The design has a simple transport layer specific for that mode (84111) that receives the samples generated by the DDS compiler in firmware and re-order the bits and gives out 16 samples every clock cycle. The results can be verified using Chipscope or using a Scope (or a spectrum analyser). 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 7.68G. For any other line rate the firmware needs to be recompiled for that specific linerate.

Following are the inputs required by the design and the outputs that user can make use of in their custom modules

Signals	Description	Direction
device_clk	reference clock from DAC38RF82	Input
mgmt_clk	100MHz internal clock	
global_rst_n	Active Low User reset	
sysref	SYSREF Signal	
Tx_sync_n	SYNC signal	
tx_serial_data	Serial Data to DAC	Output
jesd204_tx_link_data	Transport Layer data out (16 samples for every link clock)	Output from the transport module
jesd204_tx_link_valid	SOMF aligned with rx_dataout	

NOTE: As in DAC38RF82 SYNC pin is not exposed to any connector other than FMC the actual testing is performed with DAC38J82 with lane rate 7.3728G.

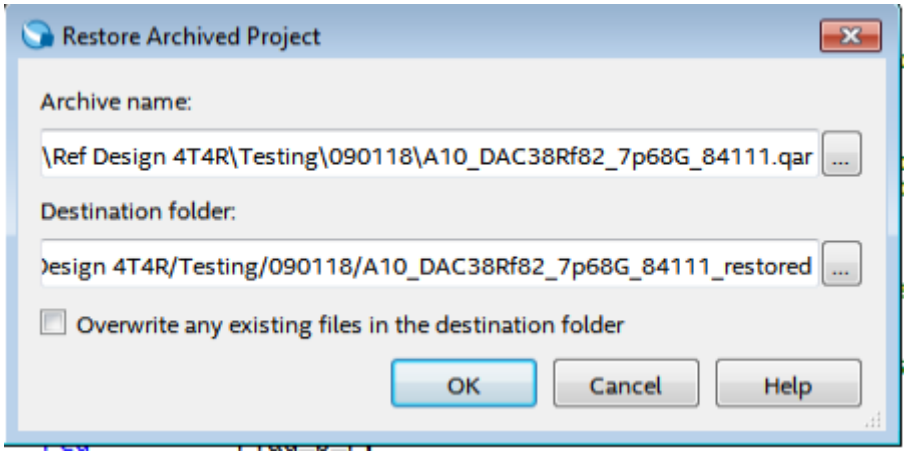
II. COMPILING QUARTUS PROJECT

1. Restore the Quartus project

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

"A10_DAC38RF82_7p68G_84111.qar"

Click OK in the dialog box that pops up next



Quartus will restore the contents to A10_DAC38RF82_7p68G_84111_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

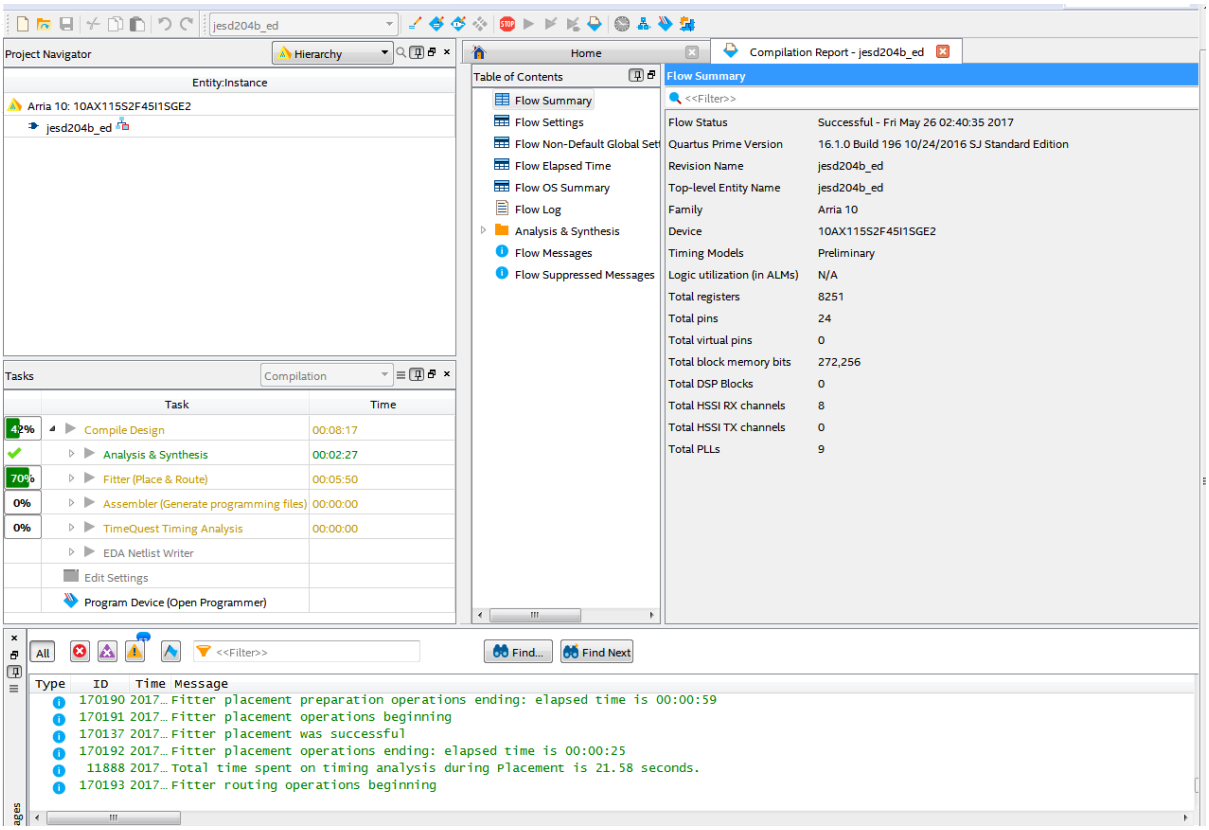
In Quartus IDE (environment), go to Project Navigator window located left side. This will show the device for which the firmware was compiled and the hierarchy of Verilog modules used.

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_DAC38RF82_7p68G_84111_restored\prj\output_files"



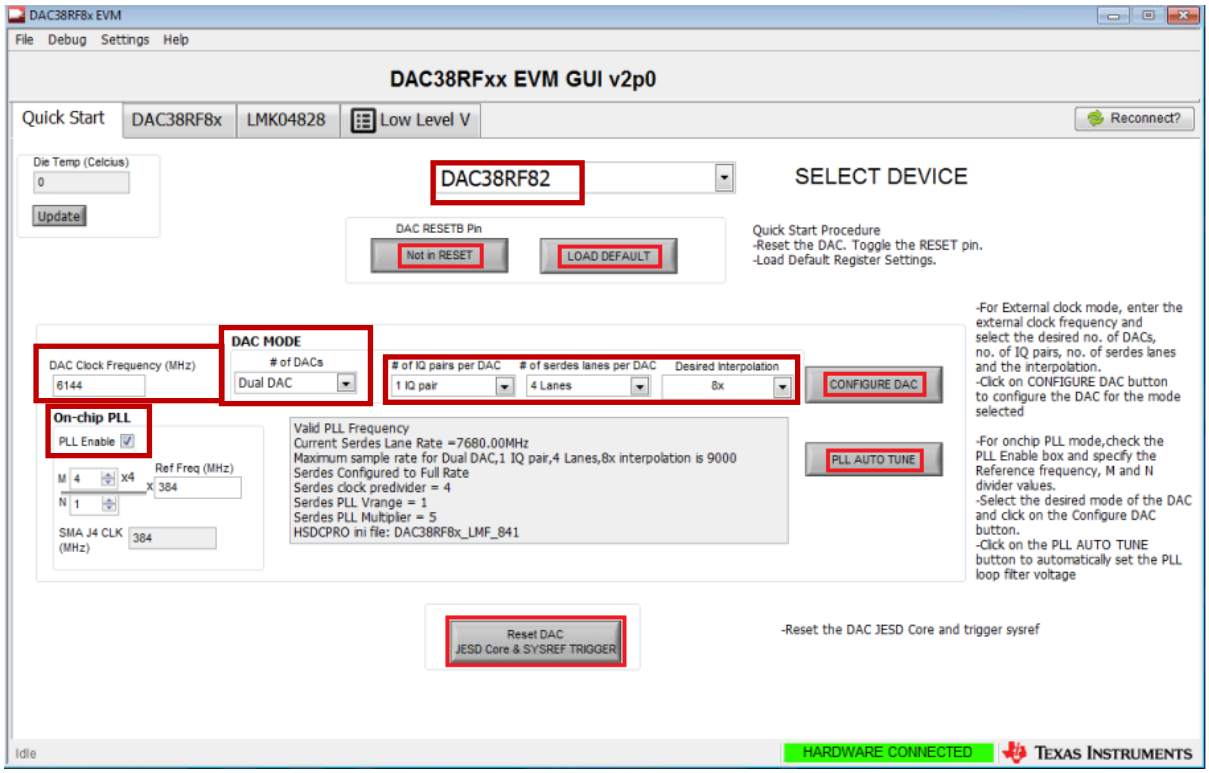
Once the SOF file is generated, proceed to next section. It will take 15 min approximately to compile the design

III. VALIDATING WITH DAC38RF82

1. Configure DAC

- The reference clock from DAC should be stable before downloading firmware. So, the DAC has to be configured first before we program the board.
- Open DAC38RF82 GUI v2p0 and perform the follow actions:
    - Reset the DAC. Toggle the Reset Pin.
    - Load the Default Register Settings.
    - Set the following values:
        - DAC Clock Frequency: 6144 Mhz
        - # of DACs : Dual DAC
        - # of IQ pairs per DAC : 1 IQ pair
        - # of serdes lanes per DAC : 4 lanes
        - Desired Interpolation : 8x
        - Check the PLL enable checkbox for On-chip PLL
    - Press the Configure DAC button.

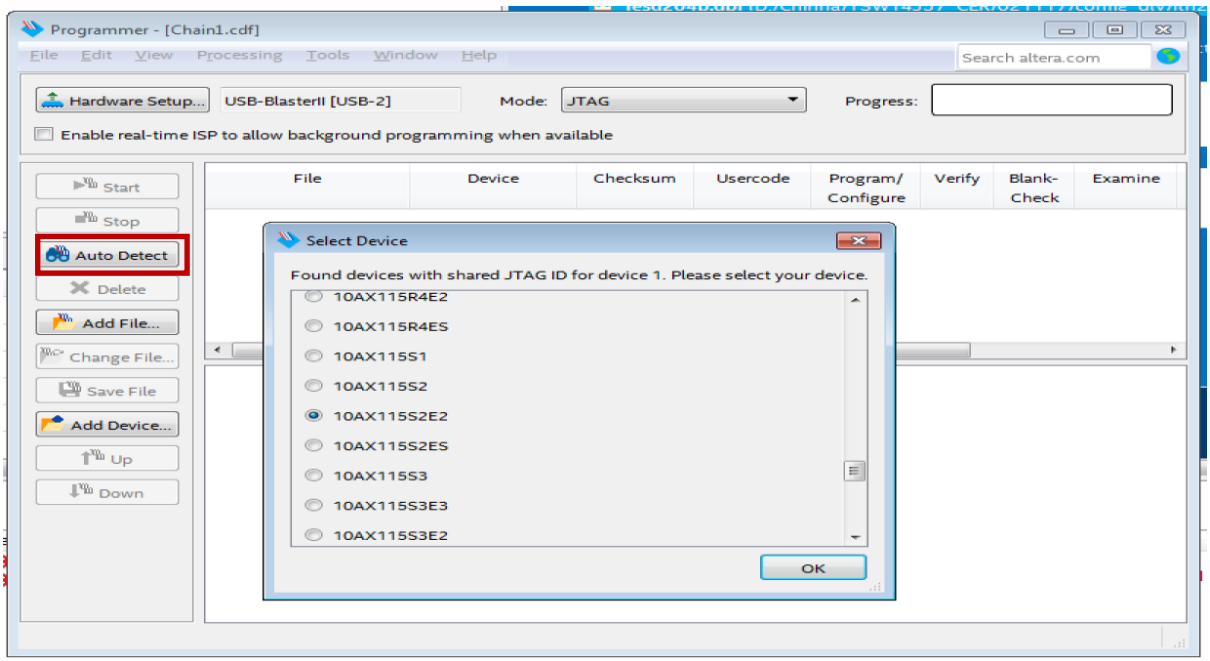
- Press on PLL Auto tune
- Finally press on Reset DAC JESD Core & SYSREF TRIGGER.



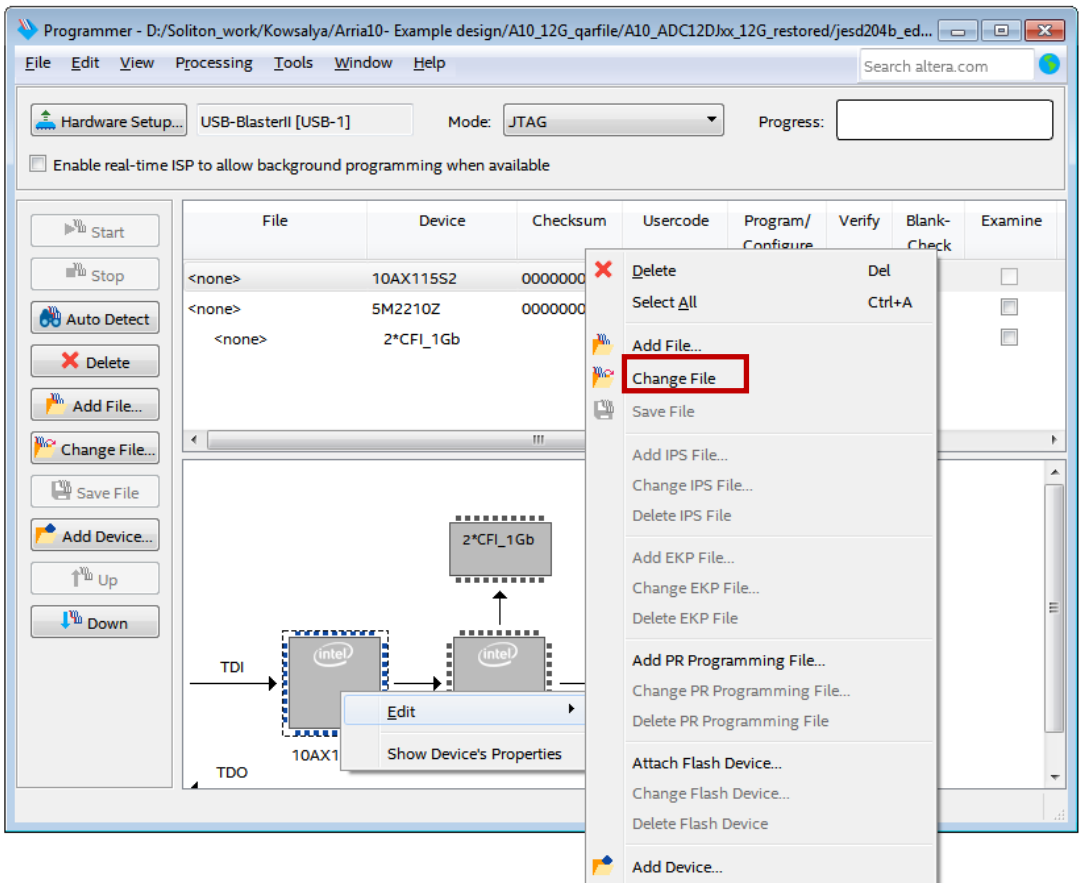
2. Programming Arria 10 development kit

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.

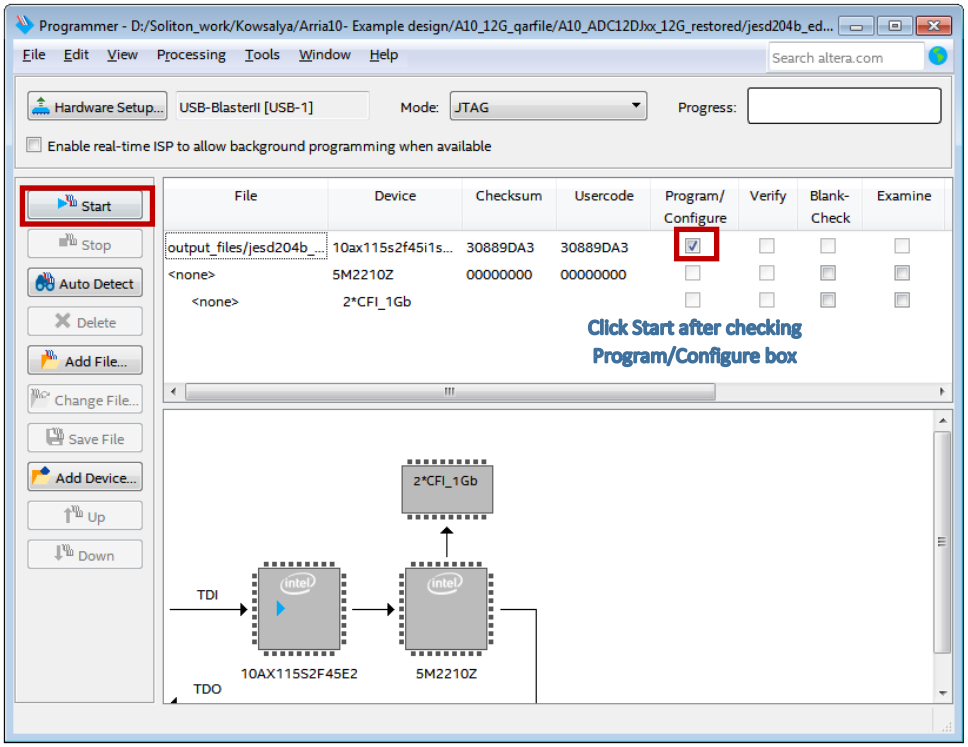
Click on Auto Detect and choose “10AX115S2E2” in the Select Device window and click OK



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

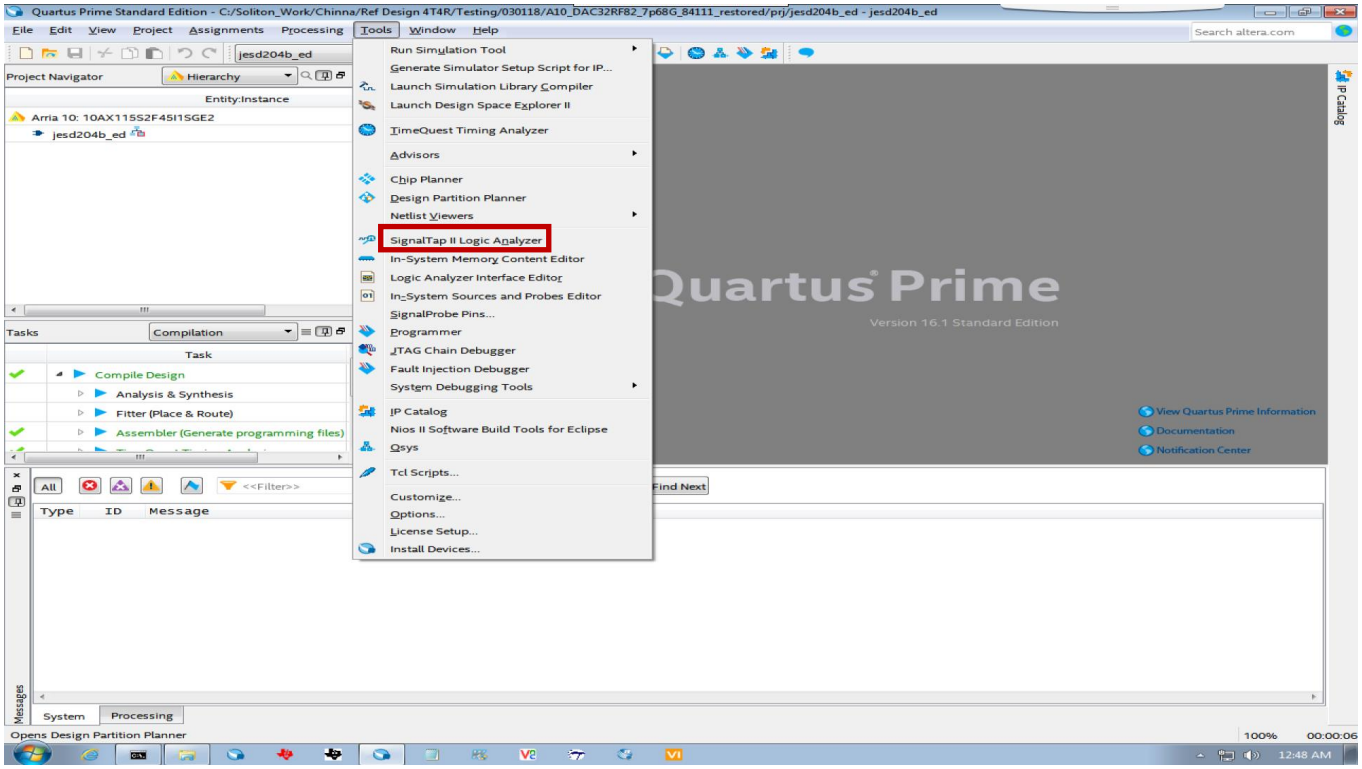


Check the Program/Configure box and click start. Progress bar should start incrementing



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

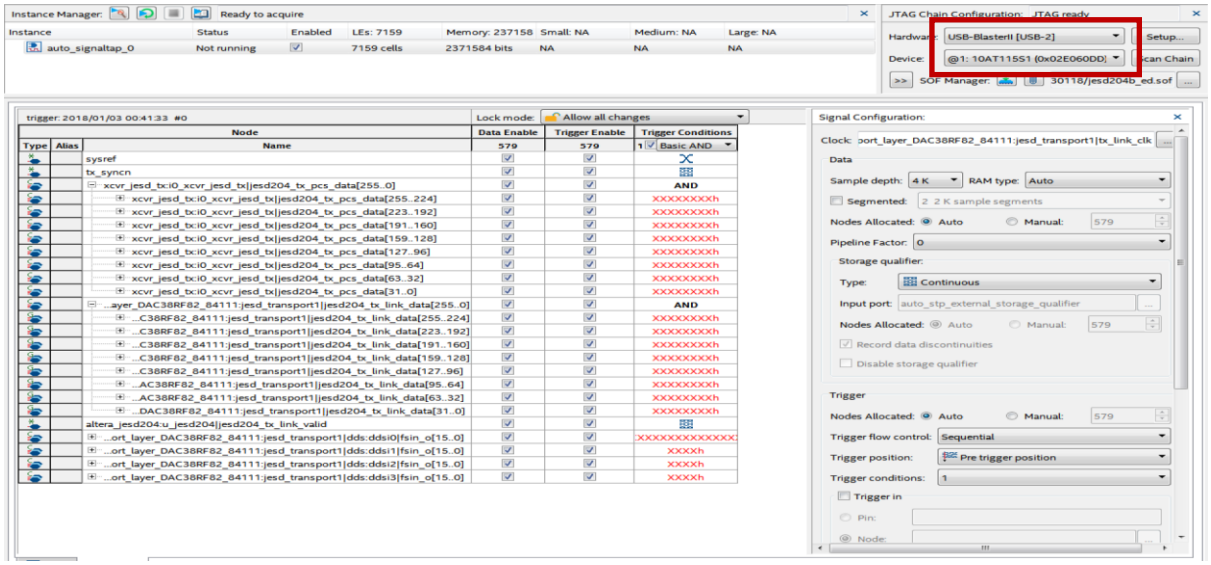


Signals which are currently probed are

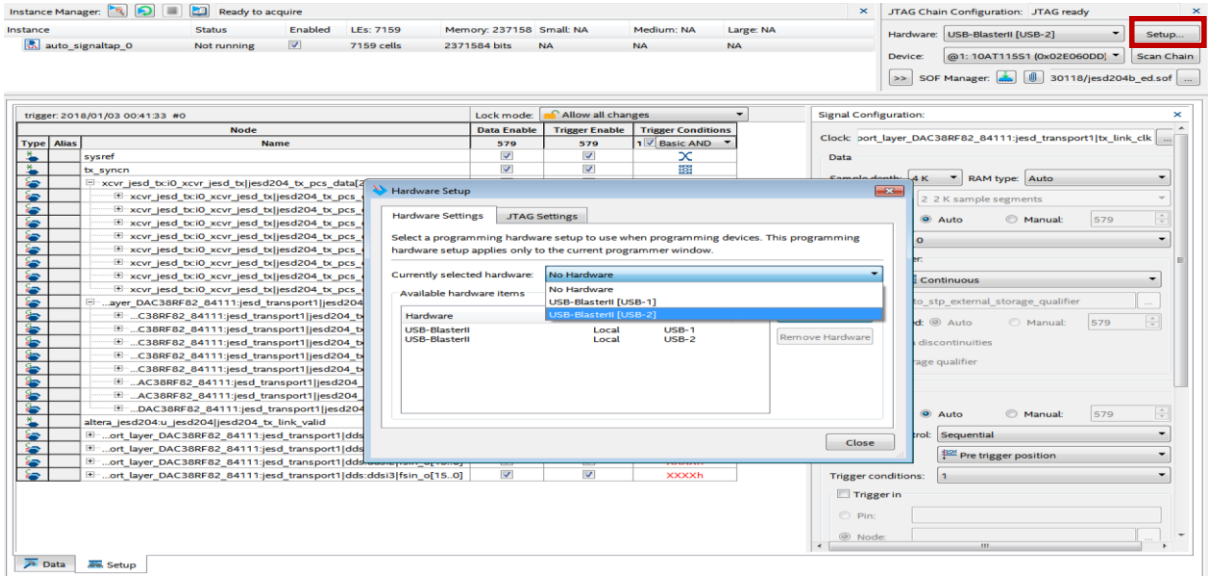
- **tx_sync_n** – Active low SYNC signal to JESD Base IP. If SYNC is established, this signal will be high
- **sysref**- SYSREF signal to Base IP
- **jesd204_tx_pcs_data** – 256 bit pcs data from JESD Base IP (grouped into 8 lanes of 32 bit each)
- **jesd204_tx_link_data** - A 256 bit bus of data coming from the Transport layer. It contains sixteen 16-bit samples in each clock cycle.
- **ddsi0|fsin_o, ddsi1|fsin_o, .. ddsi3|fsin_o**- The 16 bit output samples coming from each DDS compiler (see Firmware Information 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

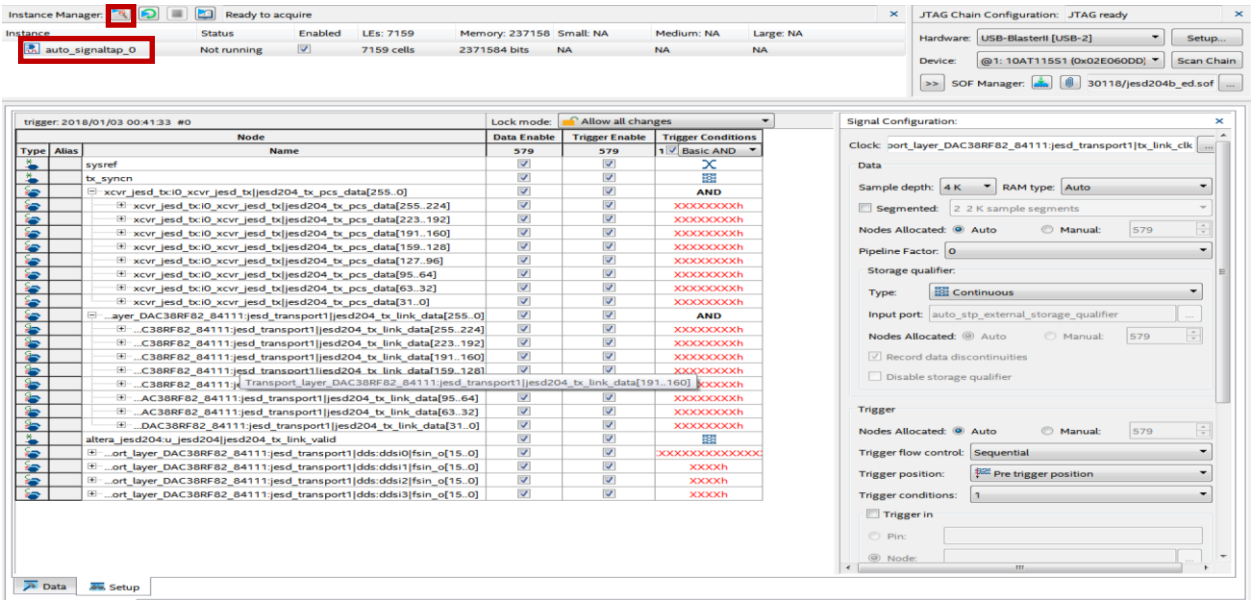
Following is the Signal Tap window. Hardware and Device tabs should be selected/listed properly



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



Select auto_signaltap_0 and Click on Run Analysis



*Refer DAC38RF82 Datasheet for more details

The transport layer is implemented only for the mode 84111

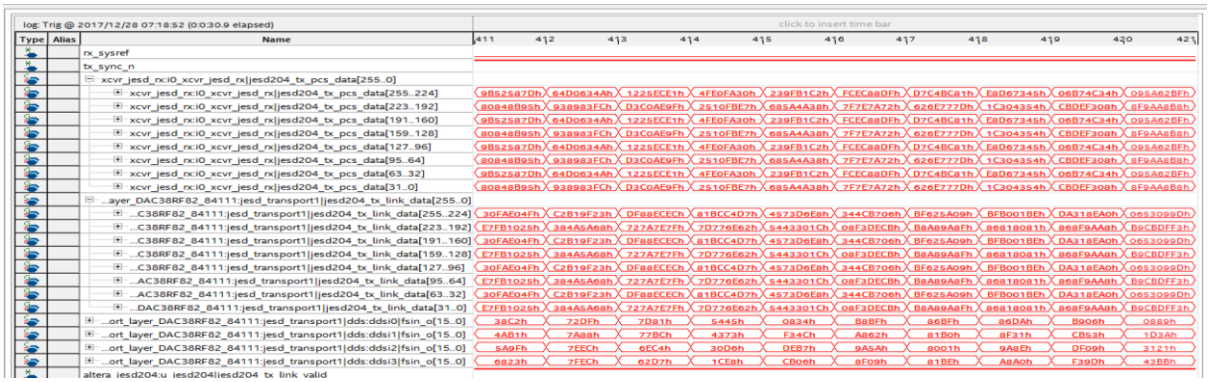
From the DAC datasheet, the sample format can be obtained as follows

Sample Pattern for one frame of 84111

Table 12. JESD204B Frame Format for LMFSHd = 84111

Nibble	1	2
lane RX0	A-i0[15:8](1)	
lane RX1	A-i0[7:0](2)	
lane RX2	A-q0[15:8]	
lane RX3	A-q0[7:0]	
lane RX4	B-i0[15:8]	
lane RX5	B-i0[7:0]	
lane RX6	B-q0[15:8]	
lane RX7	B-q0[7:0]	

STP signals for 84111, 7.68G Lane Rate

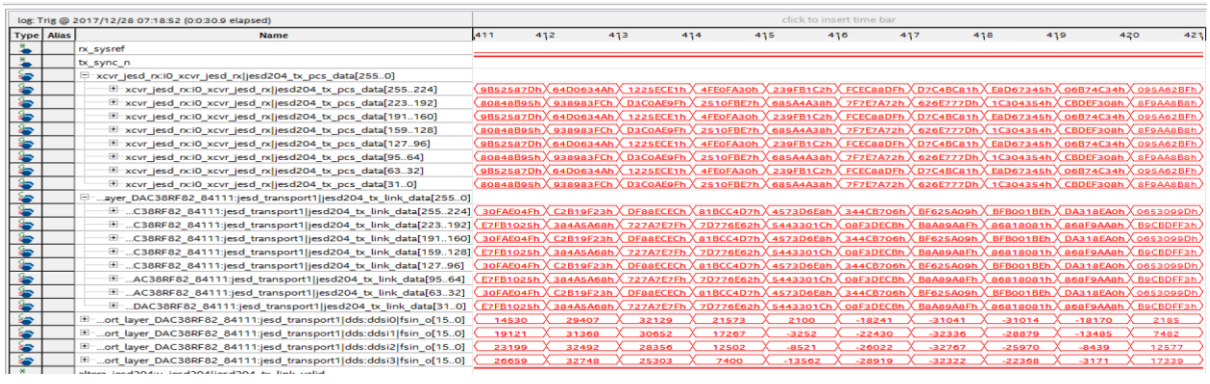


- ‘jesd204_tx_pcs_data’ shows the 256 bits of data which will be sent to DAC serially, It is grouped as 32 bit buses so that each bus corresponds to data of a single lane.
- ‘jesd204_tx_link_data’ shows the 256 bits of data which is formed by rearranging the data generated by the DDS module so that it suits 84111 mode of DDS,
  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 20 MHz) which is generated by the DDS module.

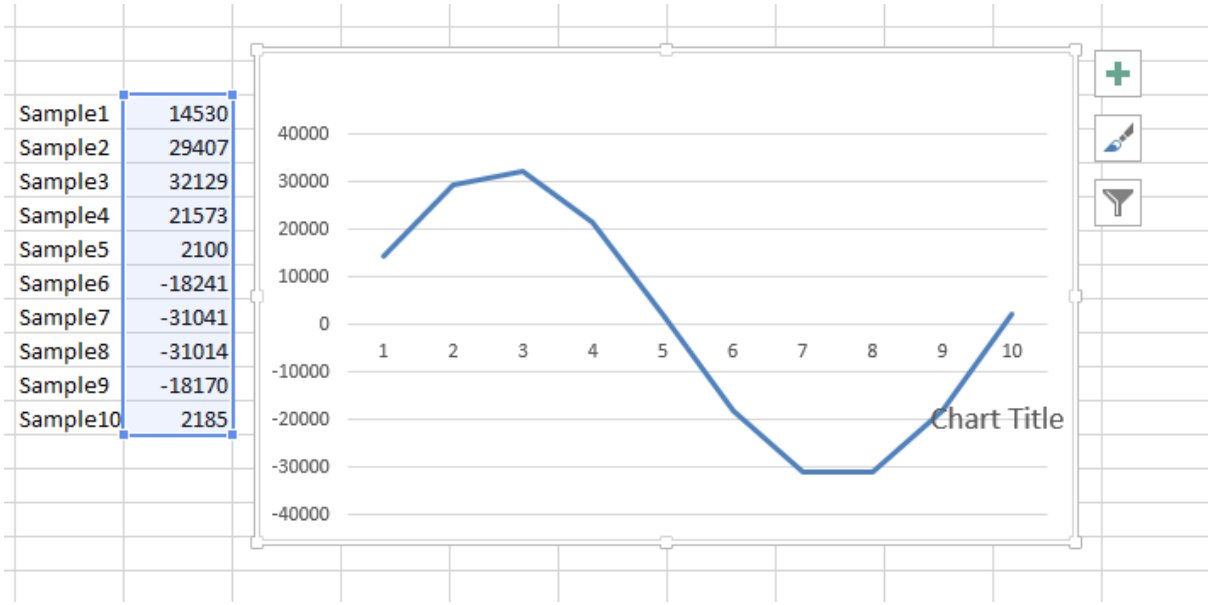
It is possible to view the output of each DDS as a 20 MHz wave, within the chipscope.

The procedure:

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

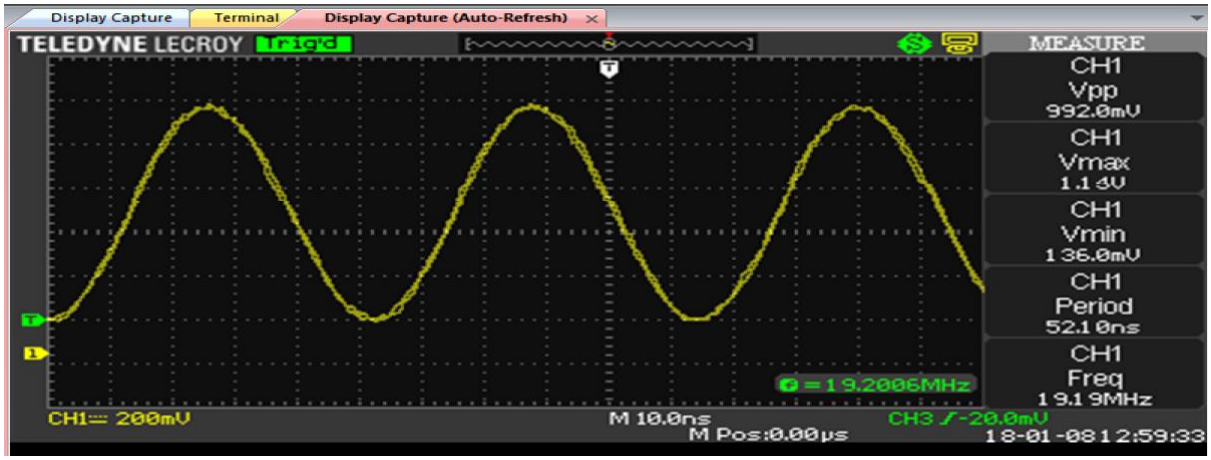


- We can plot the samples of any dds instance ouput(fsin_o) and we should get a sine wave as follows(plot of ‘ddsi0|fsin_o[15:0]’)



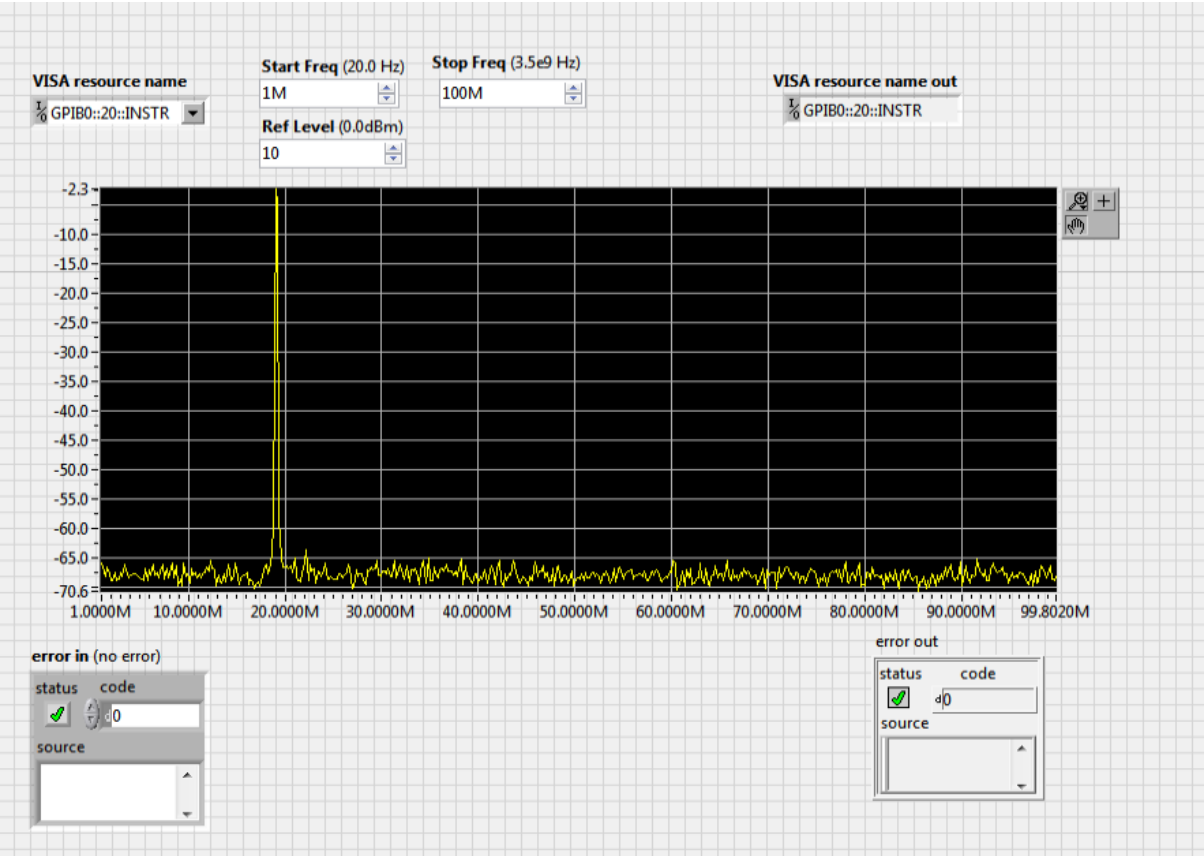
4. Scope and Spectrum Analyzer

The frequency as seen in the Scope output:



Note: As we used DAC 38J82 at lane rate 7.3728G instead of DAC38RF82 at lane rate 7.68G Sampling frequency will be 184.32 MHz instead of 192MHz, Hence the output frequency is 19.2MHz instead of 20 MHz

The frequency as seen in the Spectrum Analyser:



IV. **FIRMWARE INFORMATION**

This FW (and Transport Layer) has been made specifically for 84111 mode. A sinewave is being generated within the FW. This Sine wave 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 Sinewave. In this design, a Sine wave of frequency of 20 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 192 MHz
- For every link clock cycle we need to send 32 bits of data to the JESD Base IP. Hence we use 4 instances of DDS compiler modules to generate 4 waves of 20 MHz each. However each instance is offset by an equal value (for this mode the phase offset is one-fourth of the output wave i.e., 20/4 = 5Mhz) such that in each link clock 4

samples of a 20 Mhz Sinewave is generated. All four of these samples are concatenated (jesd204_tx_link_data signal) and given as an input to the JESD base IP.

- The concatenated output (jesd204_tx_link_data) consists of four 16-bit samples. In this mode 4 samples are divided across 2 lanes for every clock cycle. 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 (7.68G) is 768 Msps. Each DDS Compiler instance generates a 20 MHz wave w.r.t a sampling rate of 192 MHz Therefore when we combine the outputs of 4 of such instances for a single cycle, a sine wave of 20 Mhz is generated for a sampling rate of 768 Msps.
- In this design, the DDS compilers generate a free running sine wave output. Hence every clock cycle is a start of frame. Thus the output of the transport layer is directly connected to the JESD TX Base IP.

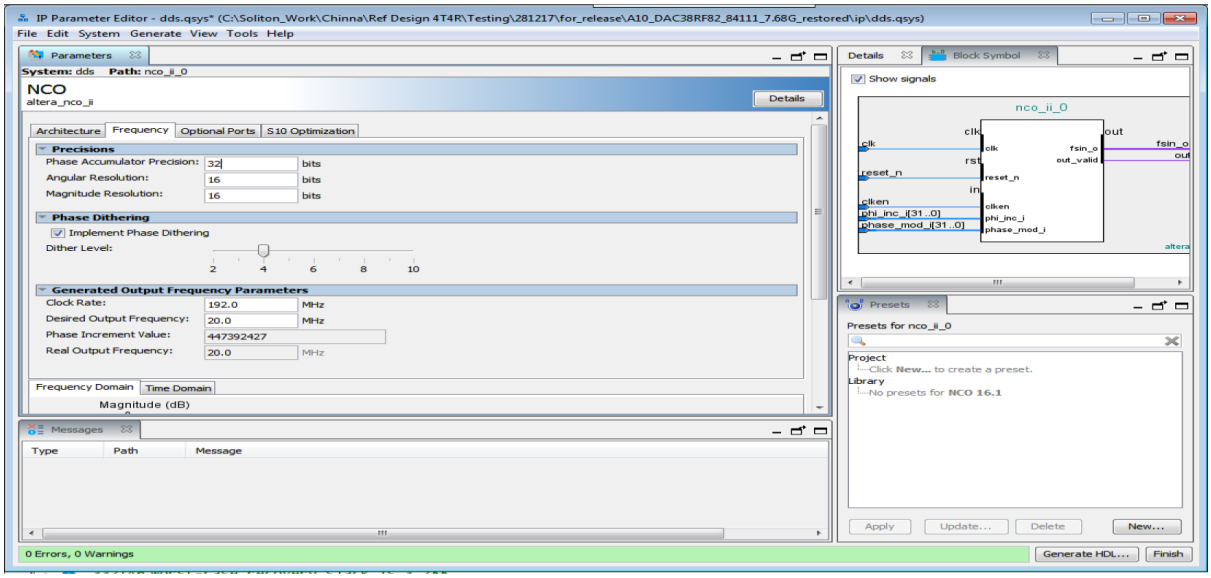
2. **Changing the Sine wave frequency**:

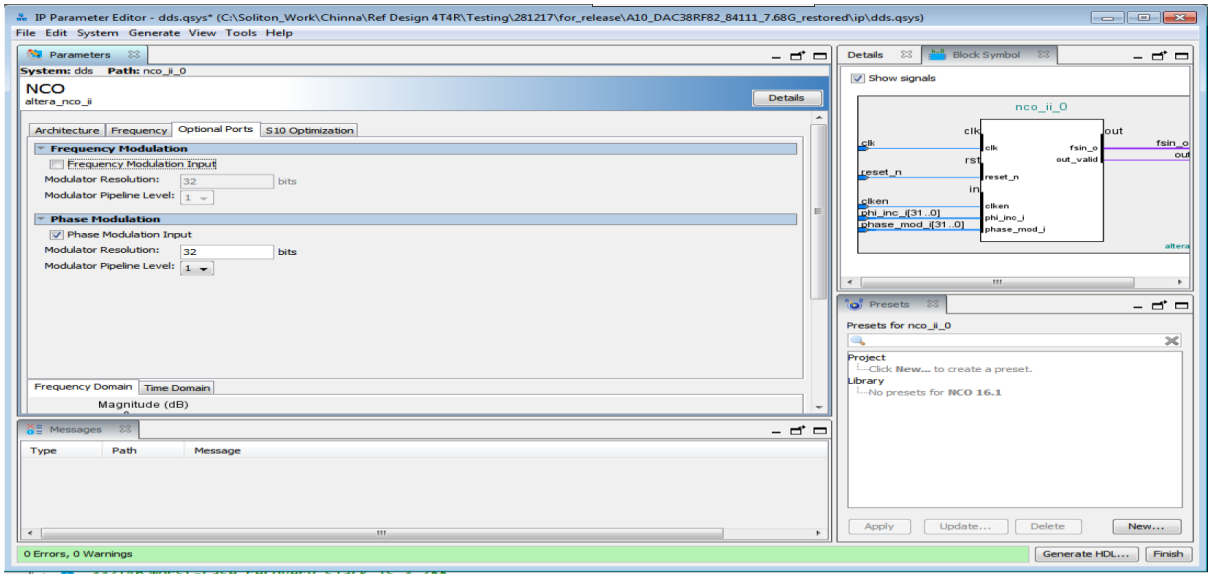
- In order to change the generated Sine wave frequency, we need to change the phase offset and phase increment values.
  (calculation has been presented for 20 MHz sine wave case)
  - Phase offset = [2^Phase width /(Sampling frequency/Required frequency)]/4
    = [2^32/(192/20)]/4
    = [4294967296/9.6]/4
    = 447392426.667/4
    = 111848106.6667
    By rounding off to the nearest integer we get 11184810707 which is 6AAAAAB in hexadecimal.

  - Phase increment = 2^Phase width /(Sampling frequency/Required frequency)
    = 2^32/(192/20)
    = 4294967296/9.6
    = 447392426.667
    By rounding off to the nearest integer and making it a multiple of phase offset, we get 447392427 which is 1AAAAAAB 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**

Two signals have been added in the top module for debugging

link_clk_led: This signal indicates if the link clock (lane rate/40 clock) from ADC is available or not. It will be blinking at the rate of about 6.4 Hz

tx_altsyncn_led: This signal refers to the SYNC input to JESD Base 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

Note: Both the LEDs are active low