General description:

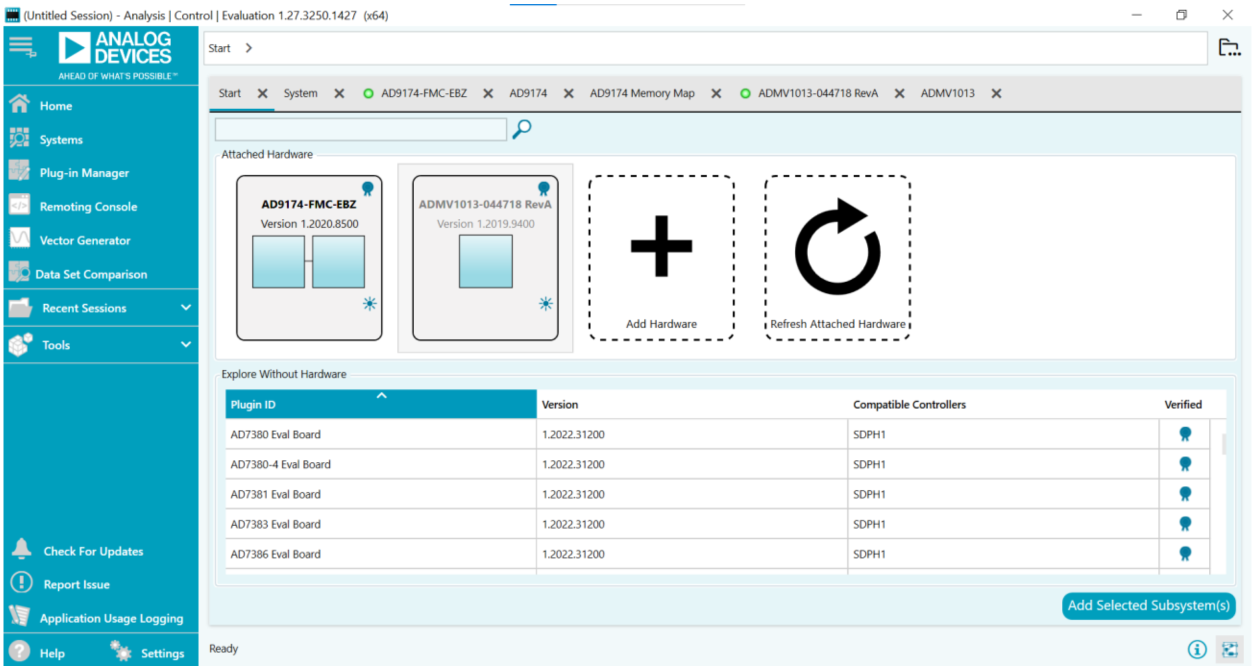
The combination of the AD9174 FMC EBZ, ADS7-V2EBZ and the ADMV 1013 can essentially generate all kinds of sequence at any frequency between 20G and 40G.

The AD9174 is a high speed, high resolution RF DAC. It has 16 bits and the sampling frequency can go up to 737.28 MHz (using internal clock). It can also do up-conversion using built-in NCOs.

The ADS7-V2EBZ is a FPGA that is used to generate the sequence in all kinds of form (single tone, QAM, etc.).

The ADMV1013 is a high frequency, broadband up-converter working from 20GHz to 40GHz.

To use this transmitter, you not only need to write the MATLAB code to generate the sequence, but also need to configure all the EVAL Boards correctly in the 'ACE' software, which is the official software suites to configure ADI products. Below is how the software looks like.



Plugin ID	Version	Compatible Controllers	Verified
AD7380 Eval Board	1.2022.31200	SDPH1	●
AD7380-4 Eval Board	1.2022.31200	SDPH1	●
AD7381 Eval Board	1.2022.31200	SDPH1	●
AD7383 Eval Board	1.2022.31200	SDPH1	●
AD7386 Eval Board	1.2022.31200	SDPH1	●

1. MATLAB code to generate sequence and communicate with FPGA

The MATLAB codes have two functions: communicate with the FPGA & DAC and generate the customized sequence.

To communicate with the FPGA and the DAC, you will need three sample transcripts provided by ADI. To generate the customized sequence, you will need to write your own MATLAB function.

1.1 MATLAB Transcripts to Communicate with Hardware

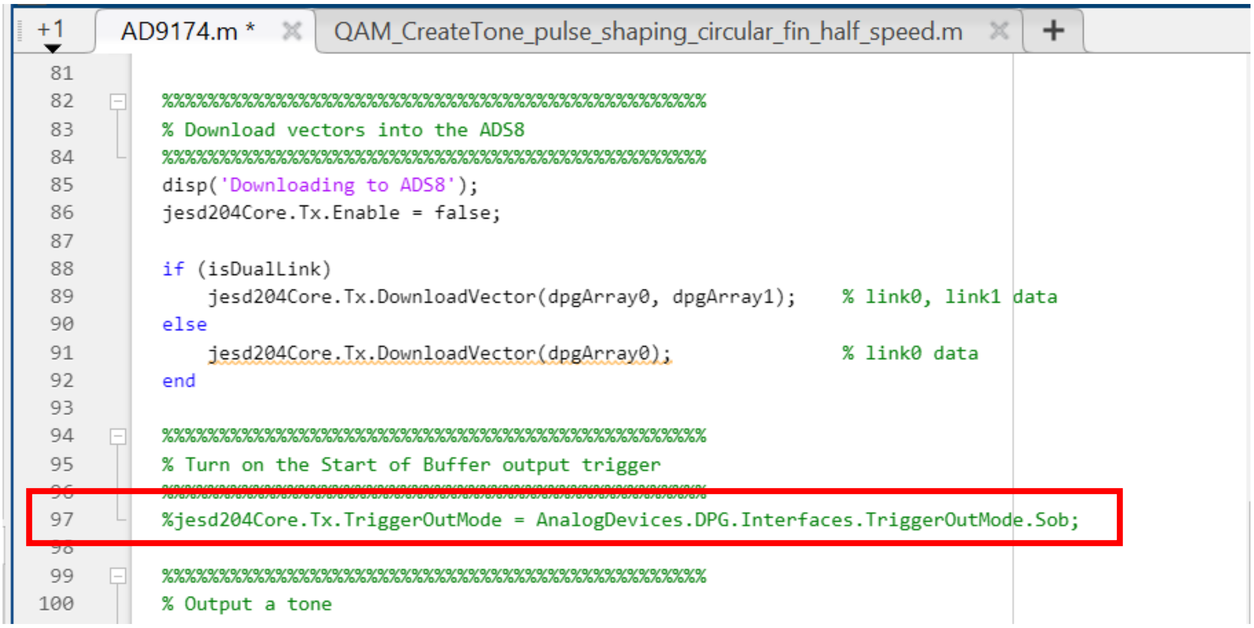
The transcripts to communicate with the hardware are shown below and you can find them in the AD9174 folder:

AD9174.m

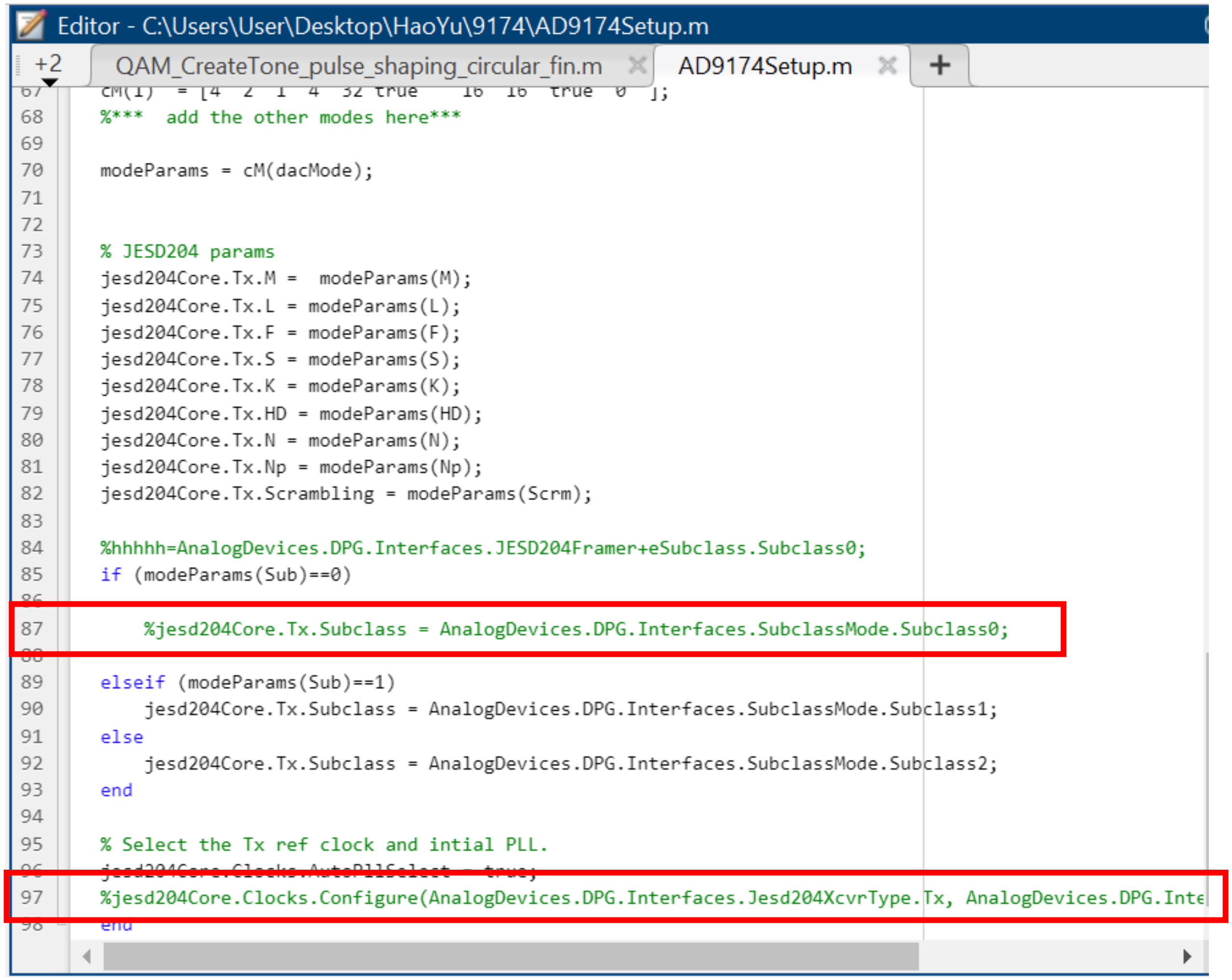
AD9174Setup.m

GetAds7.m

These sample codes have been adapted so that we can used it for the combination of AD9174 & ADS7. Unfortunately, the original sample code couldn't work unless some sentences are commented out. Below are the positions of the sentences I commented out. I totally commented out 3 sentences.



```
 81
 82    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 83    % Download vectors into the ADS8
 84    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 85    disp('Downloading to ADS8');
 86    jesd204Core.Tx.Enable = false;
 87
 88    if (isDualLink)
 89        jesd204Core.Tx.DownloadVector(dpgArray0, dpgArray1);    % link0, link1 data
 90    else
 91        jesd204Core.Tx.DownloadVector(dpgArray0);               % link0 data
 92    end
 93
 94    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 95    % Turn on the Start of Buffer output trigger
 96    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 97    %jesd204Core.Tx.TriggerOutMode = AnalogDevices.DPG.Interfaces.TriggerOutMode.Sob;
 98
 99    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
100    % Output a tone
```



Editor - C:\Users\User\Desktop\HaoYu\9174\AD9174Setup.m
+2
QAM_CreateTone_pulse_shaping_circular_fin.m
AD9174Setup.m

```
cM(1)  = [4  2  1  4  32 true    16  16  true  0  ];
%***  add the other modes here***

modeParams = cM(dacMode);


% JESD204 params
jesd204Core.Tx.M =  modeParams(M);
jesd204Core.Tx.L = modeParams(L);
jesd204Core.Tx.F = modeParams(F);
jesd204Core.Tx.S = modeParams(S);
jesd204Core.Tx.K = modeParams(K);
jesd204Core.Tx.HD = modeParams(HD);
jesd204Core.Tx.N = modeParams(N);
jesd204Core.Tx.Np = modeParams(Np);
jesd204Core.Tx.Scrambling = modeParams(Scrm);

%hhhhh=AnalogDevices.DPG.Interfaces.JESD204Framer+eSubclass.Subclass0;
if (modeParams(Sub)==0)

    %jesd204Core.Tx.Subclass = AnalogDevices.DPG.Interfaces.SubclassMode.Subclass0;

elseif (modeParams(Sub)==1)
    jesd204Core.Tx.Subclass = AnalogDevices.DPG.Interfaces.SubclassMode.Subclass1;
else
    jesd204Core.Tx.Subclass = AnalogDevices.DPG.Interfaces.SubclassMode.Subclass2;
end

% Select the Tx ref clock and intial PLL.
jesd204Core.Clocks.AutoPllSelect = true;
%jesd204Core.Clocks.Configure(AnalogDevices.DPG.Interfaces.Jesd204XcvrType.Tx, AnalogDevices.DPG.Inte
end
```

It has been proved that the code can work well even if commenting out these sentences because these sentences seem to be used to configure some parameters of the FPGA/DAC that already have their default values. You can add them back if you can find a good way to make the modification or the ADI makes a code update.

1.2 MATLAB Function to Generate Circular Sequence

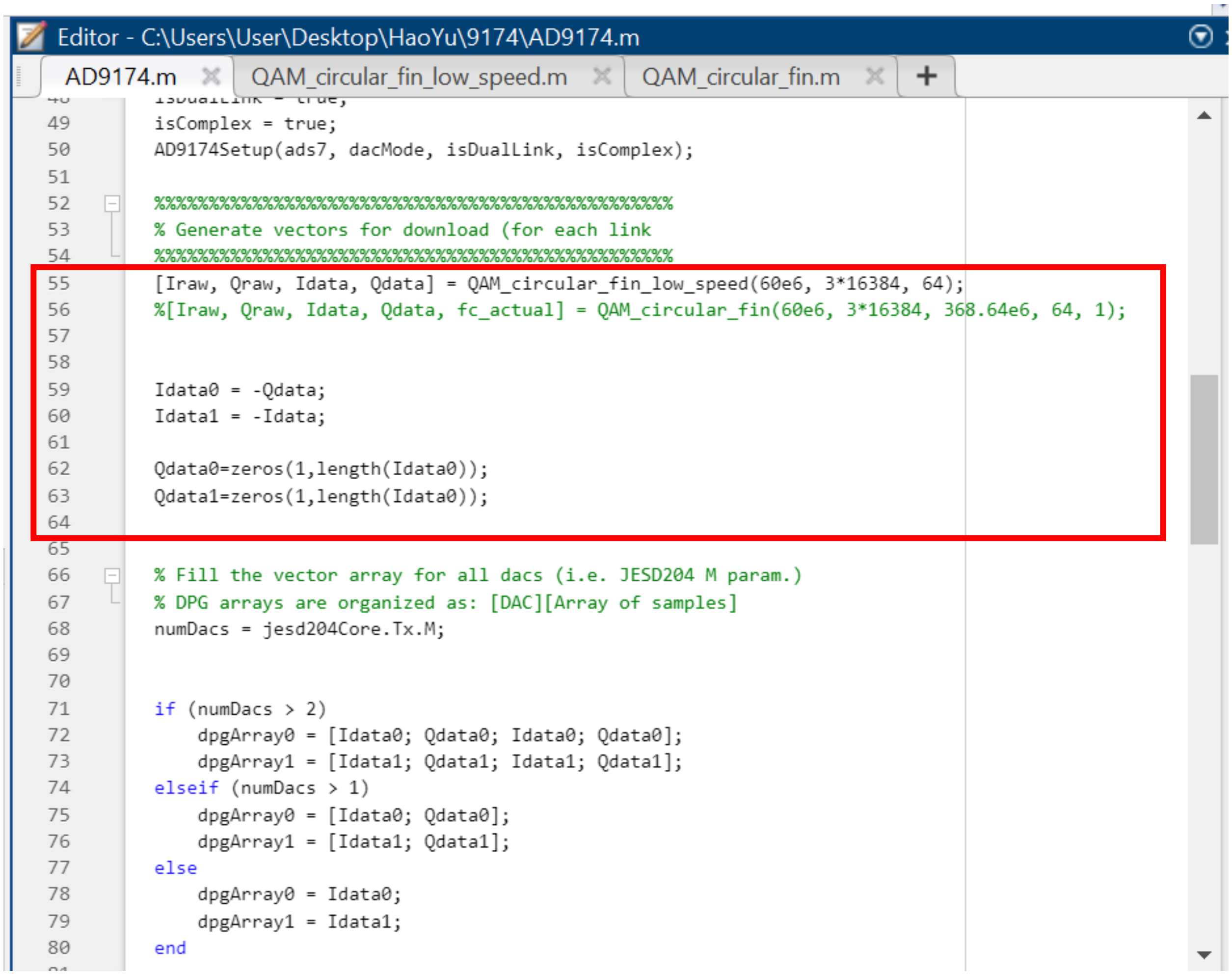
The 3 transcripts mentioned in the previous section are configured well, the only thing we need to do is to develop a MATLAB function to generate the code we need. The key thing to develop the sequence generation function, is to always make your sequence ‘**circular’** so that the FPGA can play it again and again without creating any discontinuities between beginning and the ending. There is a sample code from ADI called ‘CreateTone.m’ that demonstrates how to generate a single tone using circular sequence. You can also look at the code I wrote for the QAM for reference. The functions I wrote are called ‘QAM_circular_fin.m’ & ’QAM_circular_fin_low_speed.m’. Their usage will be detailed later.

We need to invoke the sequence generation function in the main transcript which is called ad9174.

When you finish the sequence generation function, it’s critical to assign it to the right variable so that they can be sent to the FPGA. For the DAC we are using, it has several sub-DACs for each channel. If you generate a I/Q sequence, you should send them to Idata0 & Idata1 respectively and set Qdata0 & Qdata1 as all zeros. In this case, the output sequence would be out of phase with Idata0 or Idata1, so do remember to multiply the sequence by ‘-1’ just like what I have done in the ad9174 main transcript.

The amplitude of the output data shouldn’t be too big so that it will saturate the DAC and cause non-linearity for the upconverter. You can adjust the amplitude of output data by multiplying a number smaller than 1 as I wrote in the code comment.

You should invoke the sequence generation function in the position shown below:


Editor - C:\Users\User\Desktop\HaoYu\9174\AD9174.m
AD9174.m
QAM_circular_fin_low_speed.m
QAM_circular_fin.m
+

```
isDualLink = true;
isComplex = true;
AD9174Setup(ads7, dacMode, isDualLink, isComplex);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Generate vectors for download (for each link
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[Iraw, Qraw, Idata, Qdata] = QAM_circular_fin_low_speed(60e6, 3*16384, 64);
%[Iraw, Qraw, Idata, Qdata, fc_actual] = QAM_circular_fin(60e6, 3*16384, 368.64e6, 64, 1);


Idata0 = -Qdata;
Idata1 = -Idata;

Qdata0=zeros(1,length(Idata0));
Qdata1=zeros(1,length(Idata0));


% Fill the vector array for all dacs (i.e. JESD204 M param.)
% DPG arrays are organized as: [DAC][Array of samples]
numDacs = jesd204Core.Tx.M;


if (numDacs > 2)
    dpgArray0 = [Idata0; Qdata0; Idata0; Qdata0];
    dpgArray1 = [Idata1; Qdata1; Idata1; Qdata1];
elseif (numDacs > 1)
    dpgArray0 = [Idata0; Qdata0];
    dpgArray1 = [Idata1; Qdata1];
else
    dpgArray0 = Idata0;
    dpgArray1 = Idata1;
end
```

2. Hardware connection

2.1 Power & USB connection

The FPGA uses a unique power supply. The DAC is powered by the FPGA. The ADMV 1013 up-converter uses a 5V voltage supply voltage. The maximum current for it is about 570mA when it works with the maximum gain. Connect the 5V supply to the ADMV 1013 to its 5V pin. The 3.3V & 1.8V pin are for testing only.

The FPGA, DAC & upconverter should be connected using USB cables so that you can configure them in the ACE software. When any of the EVAL Boards disconnect from the computer, reconnect them and **restart** the ACE software for a reliable re-connection.

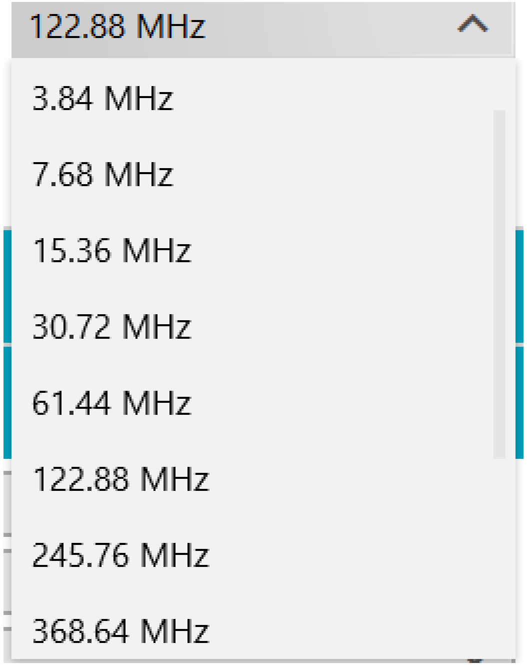
2.2 Port connection

The FPGA only needs to be connected to the DAC.

The DAC's 'DAC0' port correspond to the Idata0/Qdata0 in the MATLAB code while the 'DAC1' port corresponds to the Idata1/Qdata1. Usually, we let the DAC0 port to output in-phase data while the DAC1 port output quadrature data.

The DAC's clock can either be powered by an external source or the internal PLL. The port for external clock (J34) currently might still be damaged and cannot be used. Luckily, we do not necessarily need it. The internal PLL is by default using an internal 122.88Mhz VCO as reference. It can also accept external reference from port (J41). We can lock the internal PLL with an external reference so that the DAC clock is also locked. The port (J42) is not even mentioned in the official user manual and it hasn't been tested.

The external clock source for the PLL can only be some specific frequencies as shown below:



I would recommend using 122.88Mhz reference. The power of the reference is suggested to be 12dBm but 4dBm is also fine.

If the external clock source is disconnected, the PLL will automatically switch back to its internal clock source and will never come back. In this case, you must click 'modify' in the ACE software, then click 'Apply' to reconfigure the settings.

The ADMV 1013 upconverter can work either in I/Q (baseband) mode or IF mode. A pair of differential I/Q baseband signal is needed for the I/Q mode. However, for the DAC we are using, the biasing inductor at its output will shunt DC and low frequency components of the data to ground so we cannot use the I/Q mode. Luckily, the DAC has built-in NCOs to upconvert the baseband signal to IF. We will talk about the NCO configuration later. In this case, we will use the IF mode of the ADMV 1013.

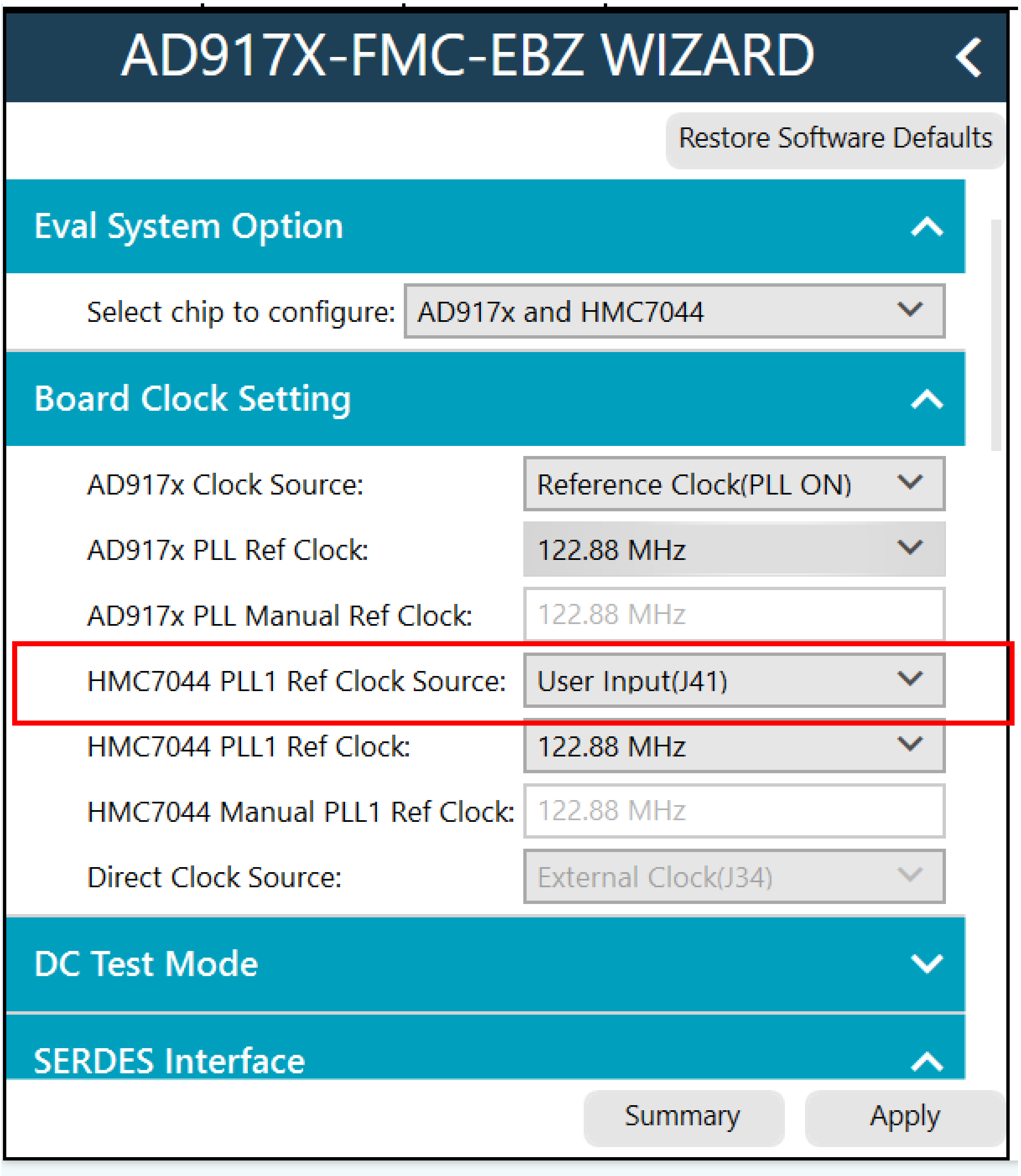
To use the ADMV 1013 in IF mode, connect the DAC0 to IF_I and DAC1 to IF_Q respectively. Since the DAC can be powered using a single phase LO, we will connect the LO signal to its LO_N or LO_P port and terminated the other with a 50-ohm load. ADMC 1013 has a built-in quadrupler which means the IF signal from the DAC will be mixed with 4x the frequency you connect to the LO port. The input LO power should be with -6dBm to 6dBm.

Finally, connect the RF_OUT port to the spectrum analyzer or the DUT.

3. ACE configuration

3.1 FPGA & DAC configuration

Configure the DAC/FPGA in the ACE software as shown below, you only need to set using external PLL reference (already marked in red rectangular). The others should be left as default.



Select chip to configure: AD917x and HMC7044

AD917x Clock Source: Reference Clock(PLL ON)

AD917x PLL Ref Clock: 122.88 MHz

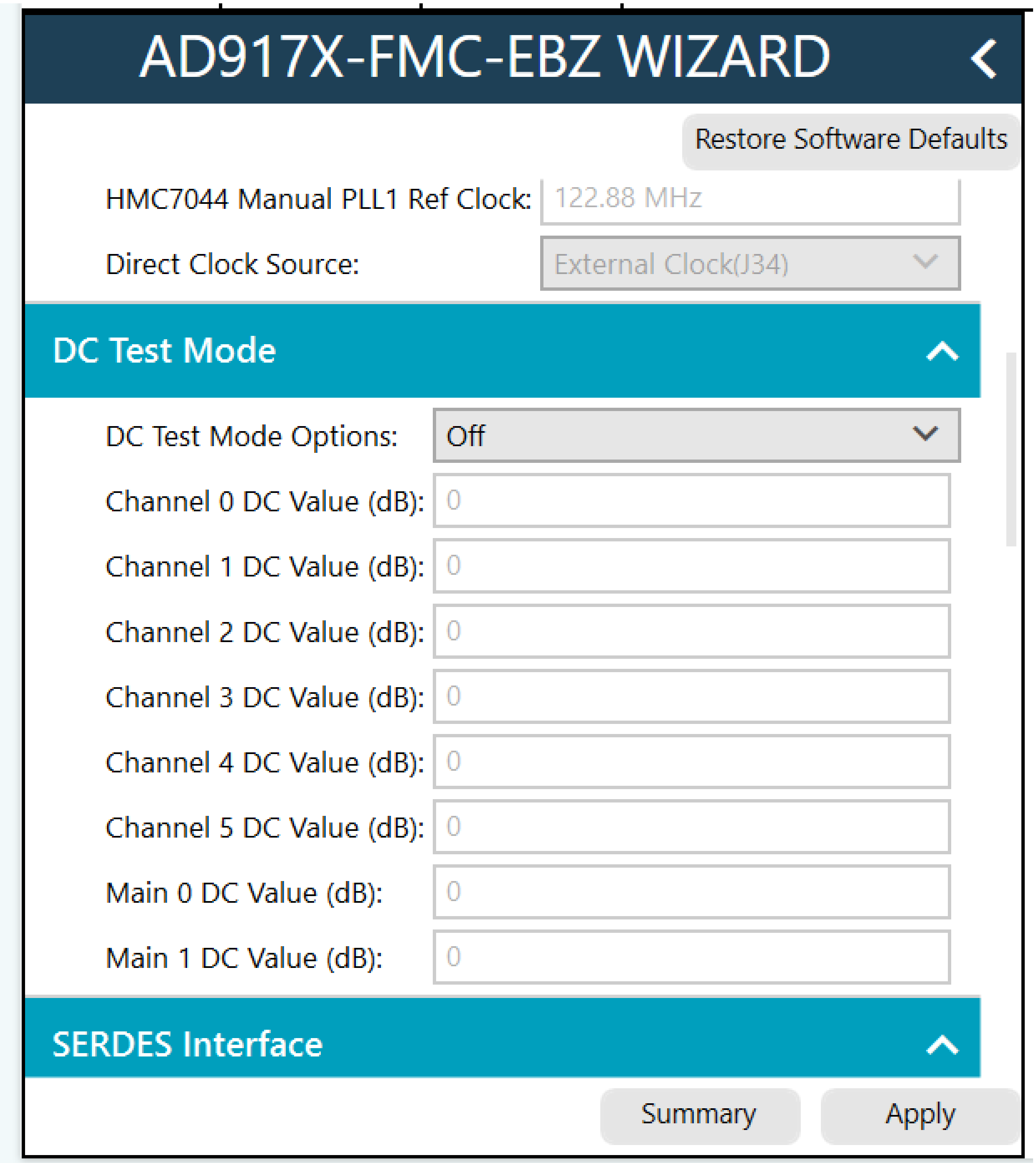
AD917x PLL Manual Ref Clock: 122.88 MHz

HMC7044 PLL1 Ref Clock Source: User Input(J41)

HMC7044 PLL1 Ref Clock: 122.88 MHz

HMC7044 Manual PLL1 Ref Clock: 122.88 MHz

Direct Clock Source: External Clock(J34)



AD917X-FMC-EBZ WIZARD

HMC7044 Manual PLL1 Ref Clock: 122.88 MHz

Direct Clock Source: External Clock(J34)

DC Test Mode

DC Test Mode Options: Off

Channel 0 DC Value (dB): 0

Channel 1 DC Value (dB): 0

Channel 2 DC Value (dB): 0

Channel 3 DC Value (dB): 0

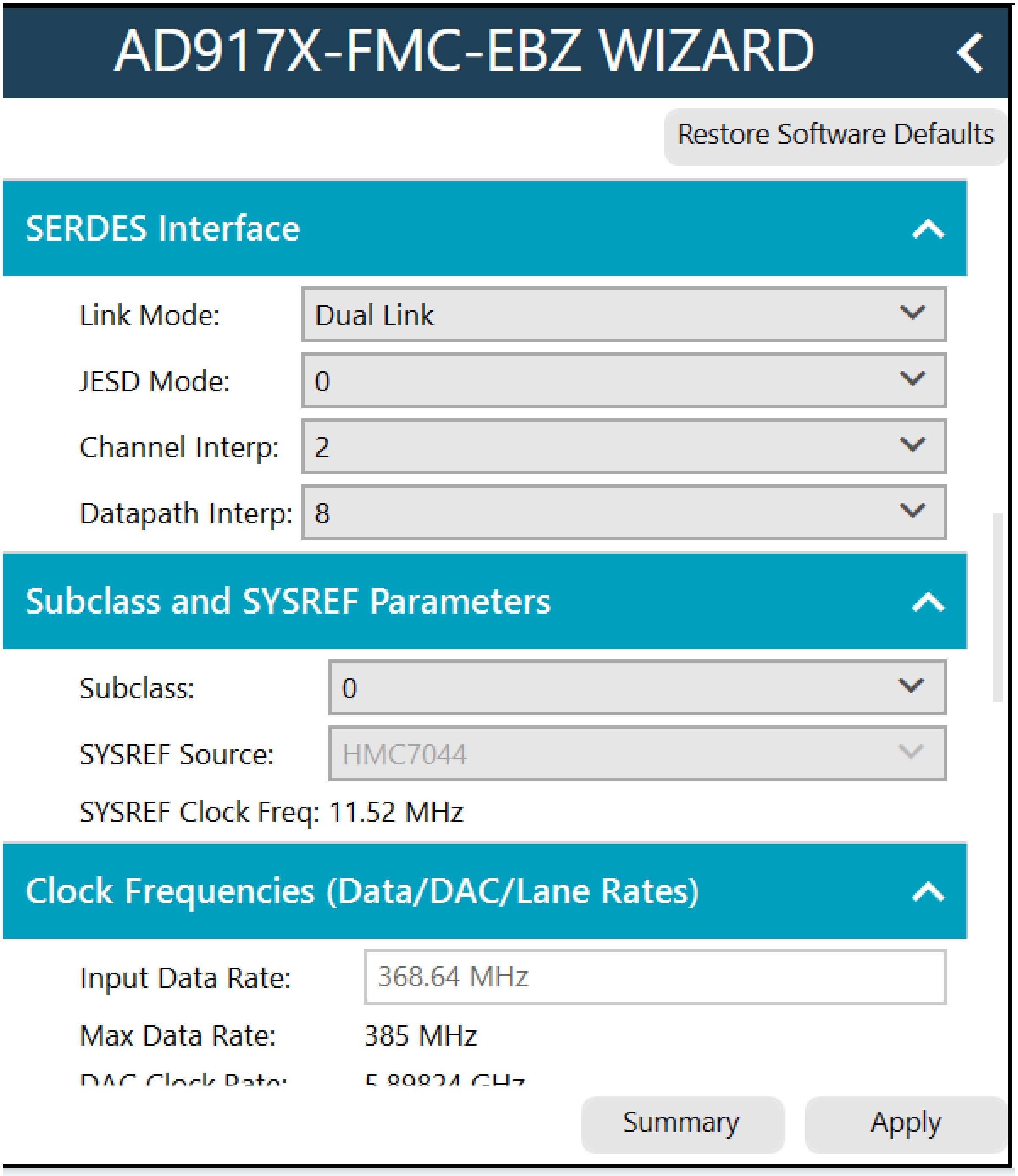
Channel 4 DC Value (dB): 0

Channel 5 DC Value (dB): 0

Main 0 DC Value (dB): 0

Main 1 DC Value (dB): 0

SERDES Interface



AD917X-FMC-EBZ WIZARD

SERDES Interface

Link Mode: Dual Link

JESD Mode: 0

Channel Interp: 2

Datapath Interp: 8

Subclass and SYSREF Parameters

Subclass: 0

SYSREF Source: HMC7044

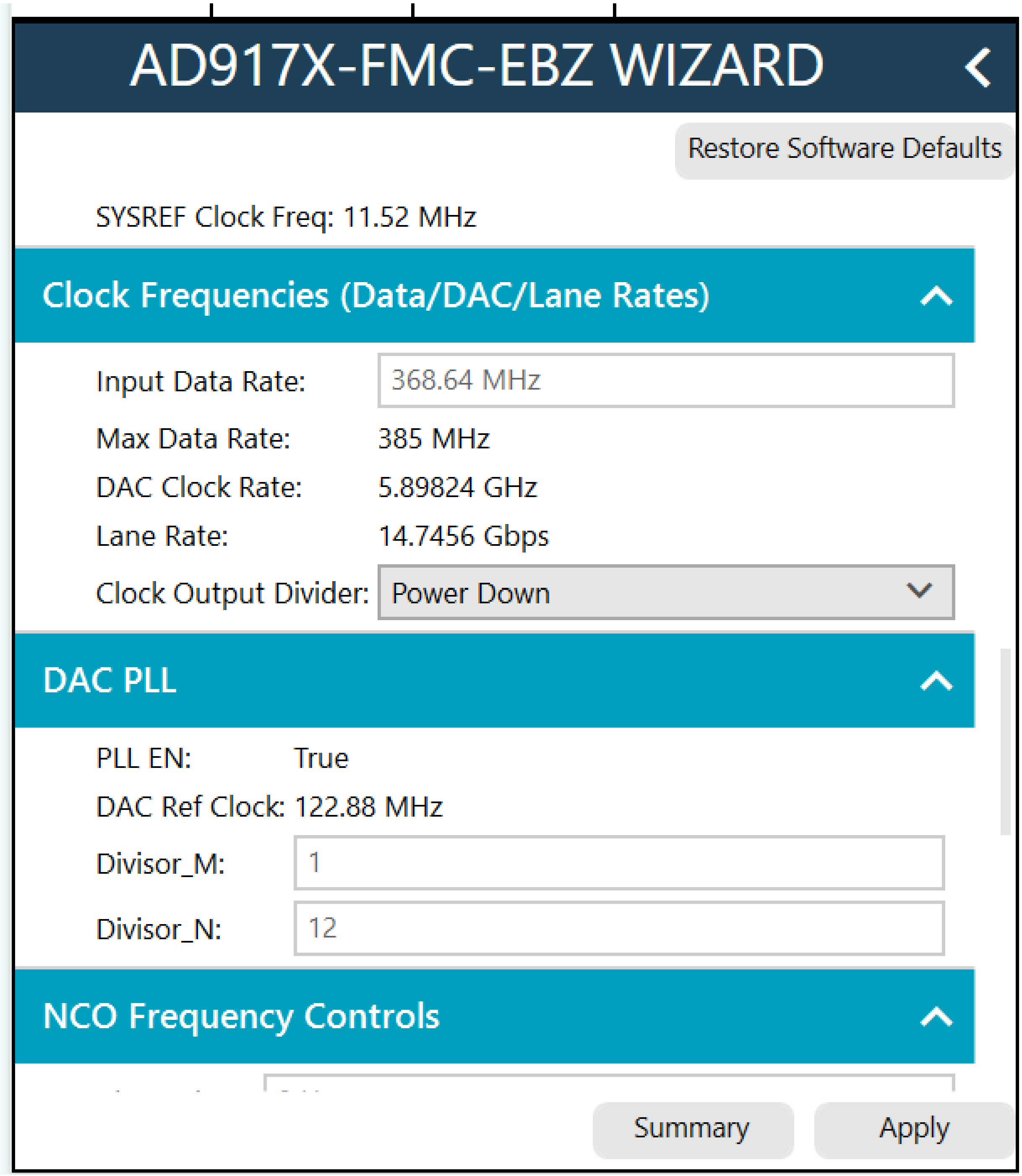
SYSREF Clock Freq: 11.52 MHz

Clock Frequencies (Data/DAC/Lane Rates)

Input Data Rate: 368.64 MHz

Max Data Rate: 385 MHz

DAC Clock Rate: 5.89824 GHz



AD917X-FMC-EBZ WIZARD

SYSREF Clock Freq: 11.52 MHz

Clock Frequencies (Data/DAC/Lane Rates)

Input Data Rate: 368.64 MHz

Max Data Rate: 385 MHz

DAC Clock Rate: 5.89824 GHz

Lane Rate: 14.7456 Gbps

Clock Output Divider: Power Down

DAC PLL

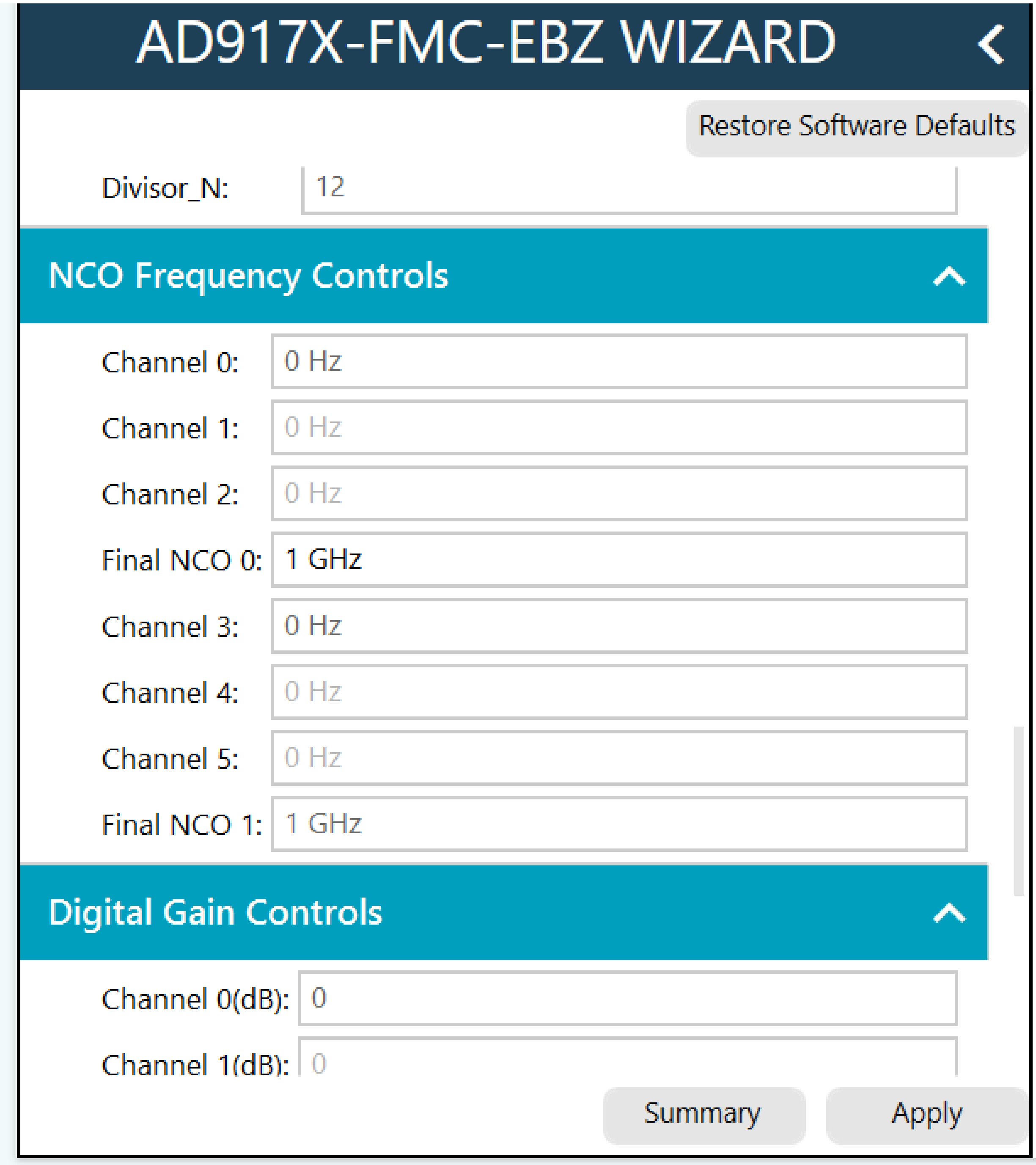
PLL EN: True

DAC Ref Clock: 122.88 MHz

Divisor_M: 1

Divisor_N: 12

NCO Frequency Controls



Divisor_N: 12

NCO Frequency Controls

Channel 0: 0 Hz

Channel 1: 0 Hz

Channel 2: 0 Hz

Final NCO 0: 1 GHz

Channel 3: 0 Hz

Channel 4: 0 Hz

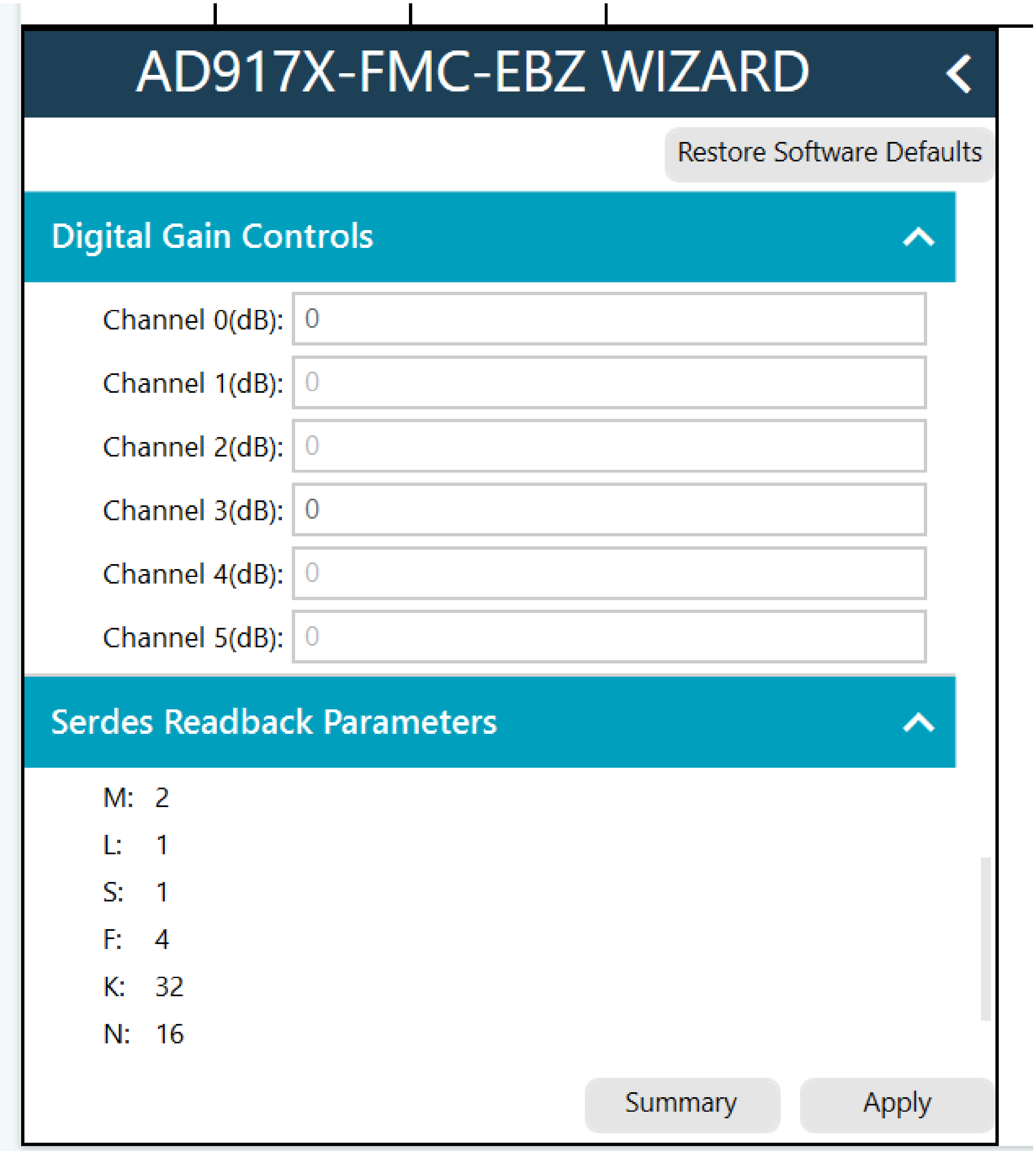
Channel 5: 0 Hz

Final NCO 1: 1 GHz

Digital Gain Controls

Channel 0(dB): 0

Channel 1(dB): 0



AD917X-FMC-EBZ WIZARD

Digital Gain Controls

Channel 0(dB): 0

Channel 1(dB): 0

Channel 2(dB): 0

Channel 3(dB): 0

Channel 4(dB): 0

Channel 5(dB): 0

Serdes Readback Parameters

M: 2

L: 1

S: 1

F: 4

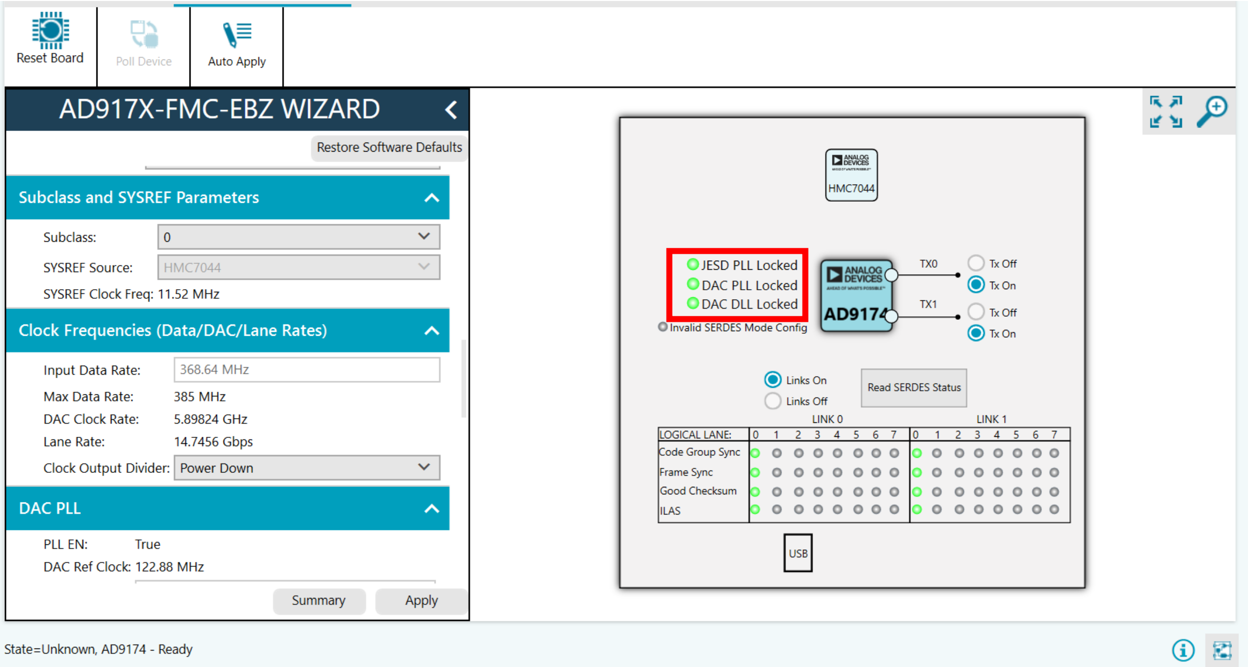
K: 32

N: 16

There are only a few things to be noticed.

First, it's recommended to use the default 368.64MHz DAC clock frequency. It's recommended to use 1GHz for the NCO to generate a 1GHz IF signal because the Upconverter has best overall performance with a 1GHz IF input.

When you finish the configuration, click 'Apply'. When you see all the green circles, it means the DAC & FPGA is running well.



Subclass: 0

SYSREF Source: HMC7044

SYSREF Clock Freq: 11.52 MHz

Input Data Rate: 368.64 MHz

Max Data Rate: 385 MHz

DAC Clock Rate: 5.89824 GHz

Lane Rate: 14.7456 Gbps

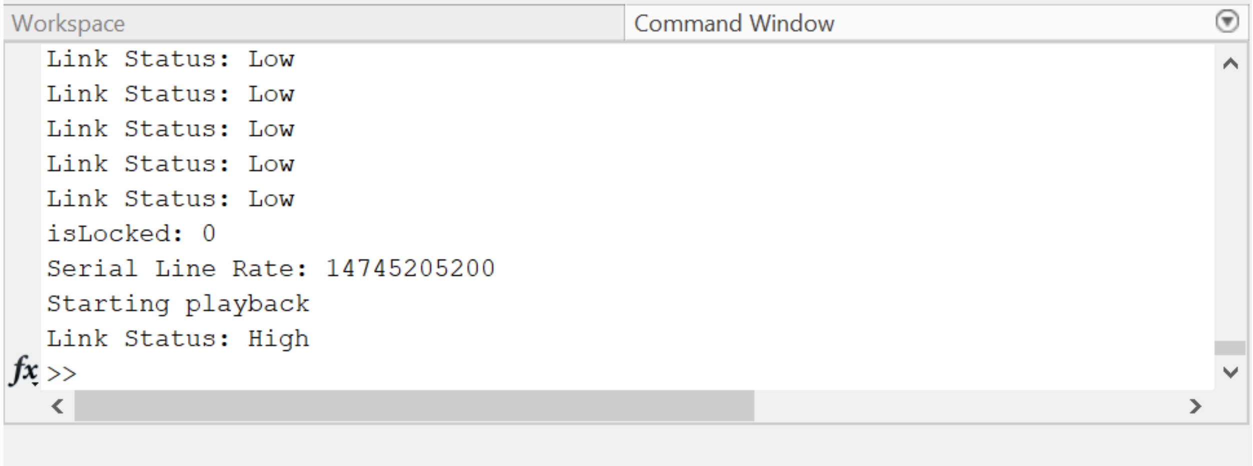
Clock Output Divider: Power Down

PLL EN: True

DAC Ref Clock: 122.88 MHz

State=Unknown, AD9174 - Ready

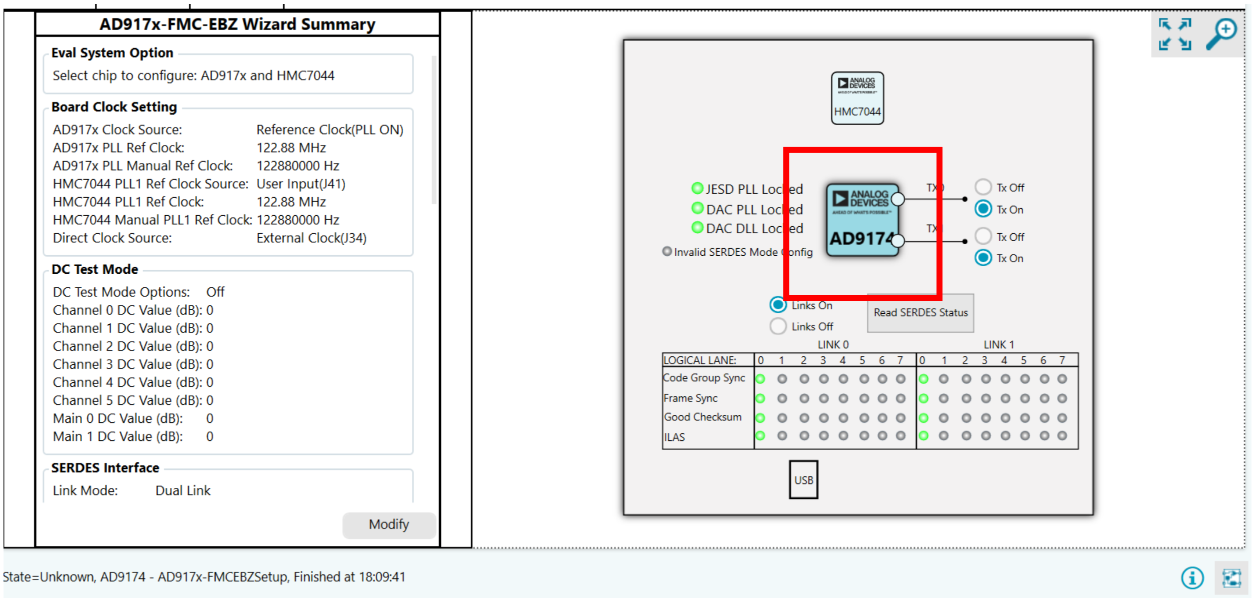
After you've set up the ACE software, open ad9174.m in MATLAB and run the transcript. Click 'OK' for any window prompted. When the Link Status is high, the DAC is successfully outputting data.



```
Link Status: Low
Link Status: Low
Link Status: Low
Link Status: Low
Link Status: Low
isLocked: 0
Serial Line Rate: 14745205200
Starting playback
Link Status: High
fx >>
```

We also need to set up the phase offset for the NCO so that the NCO at the Quadrature channel has a 90-degree phase offset compared to the in-phase channel.

Double click the 'ad9174' symbol, you will then see its schematic. Then click 'Proceed to Memory Map'.



AD917x-FMC-EBZ Wizard Summary

Eval System Option

Select chip to configure: AD917x and HMC7044

Board Clock Setting

AD917x Clock Source:	Reference Clock(PLL ON)
AD917x PLL Ref Clock:	122.88 MHz
AD917x PLL Manual Ref Clock:	122880000 Hz
HMC7044 PLL1 Ref Clock Source:	User Input(J41)
HMC7044 PLL1 Ref Clock:	122.88 MHz
HMC7044 Manual PLL1 Ref Clock:	122880000 Hz
Direct Clock Source:	External Clock(J34)

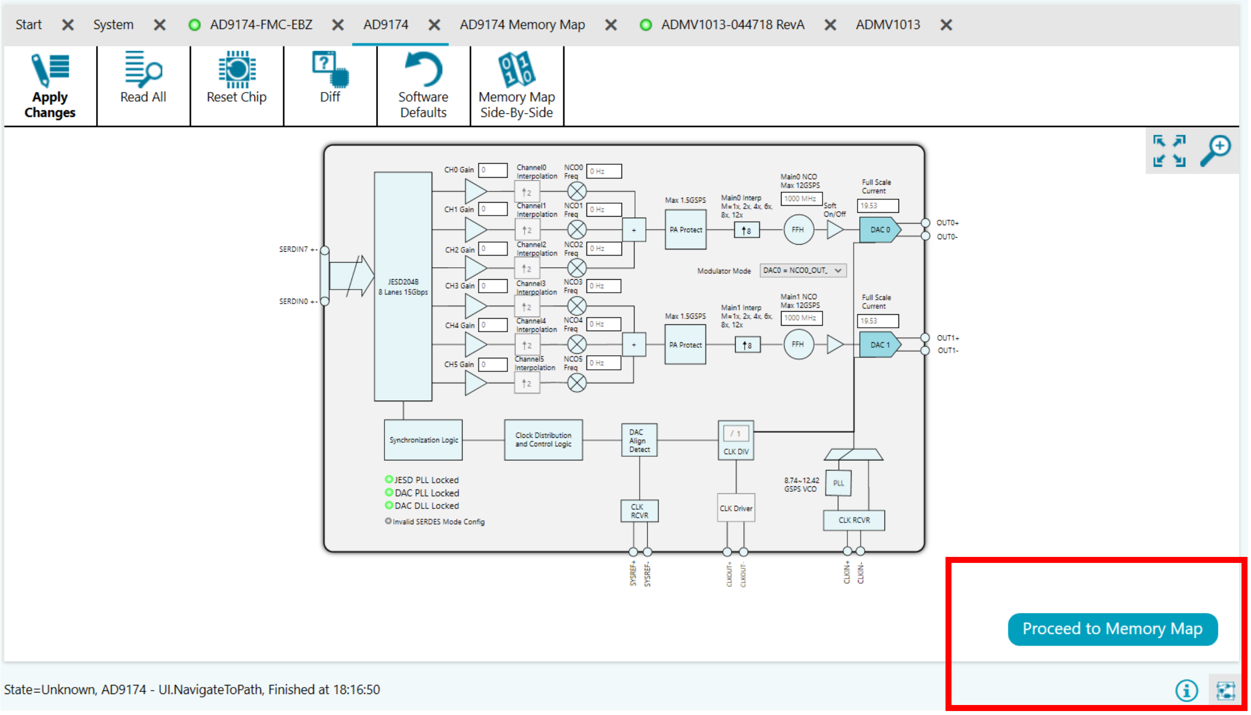
DC Test Mode

DC Test Mode Options:	Off
Channel 0 DC Value (dB):	0
Channel 1 DC Value (dB):	0
Channel 2 DC Value (dB):	0
Channel 3 DC Value (dB):	0
Channel 4 DC Value (dB):	0
Channel 5 DC Value (dB):	0
Main 0 DC Value (dB):	0
Main 1 DC Value (dB):	0

SERDES Interface

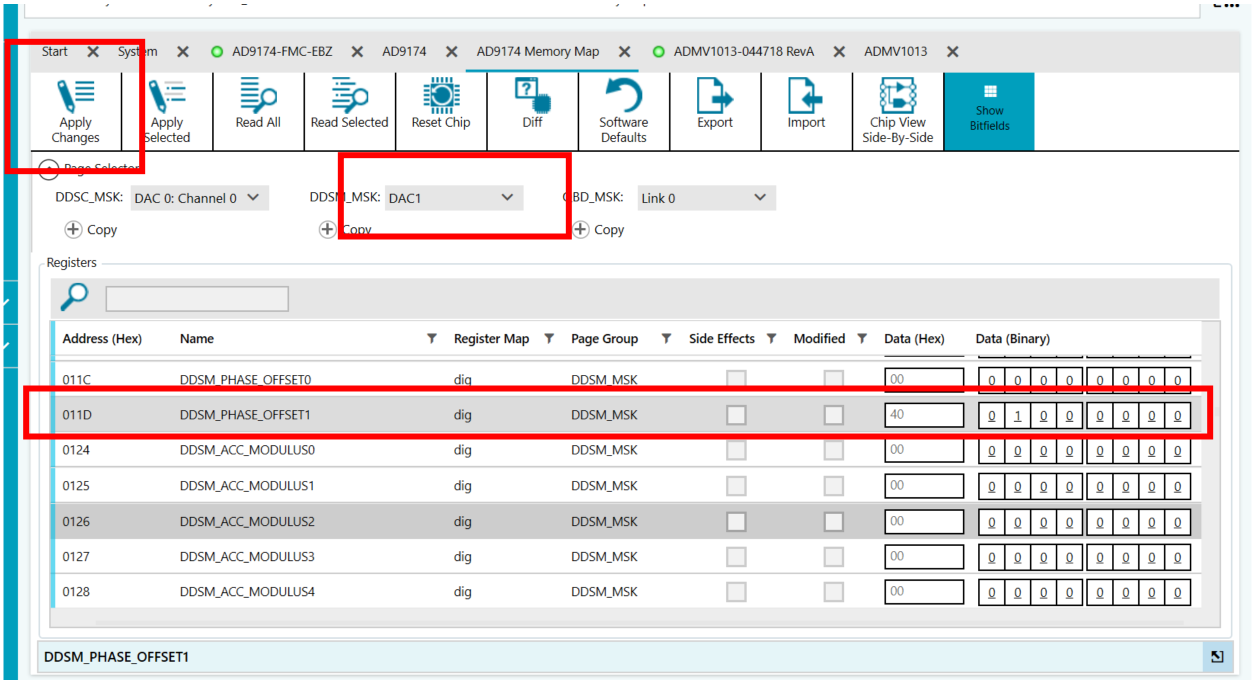
Link Mode: Dual Link

State=Unknown, AD9174 - AD917x-FMCEBZSetup, Finished at 18:09:41



State=Unknown, AD9174 - UI.NavigateToPath, Finished at 18:16:50

Select DAC1 and then change x011D from 00(HEX) to 40(HEX). This represent 90-degree phase offset. You can look at the AD9174 user guide for more detail. Then click ‘Apply Changes’.



DDSC_MSK: DAC 0: Channel 0

DDSM_MSK: DAC1

BD_MSK: Link 0

Address (Hex)	Name	Register Map	Page Group	Side Effects	Modified	Data (Hex)	Data (Binary)
011C	DDSM_PHASE_OFFSET0	dig	DDSM_MSK	□	□	00	0 0 0 0 0 0 0 0
011D	DDSM_PHASE_OFFSET1	dig	DDSM_MSK	□	□	40	0 1 0 0 0 0 0 0
0124	DDSM_ACC_MODULUS0	dig	DDSM_MSK	□	□	00	0 0 0 0 0 0 0 0
0125	DDSM_ACC_MODULUS1	dig	DDSM_MSK	□	□	00	0 0 0 0 0 0 0 0
0126	DDSM_ACC_MODULUS2	dig	DDSM_MSK	□	□	00	0 0 0 0 0 0 0 0
0127	DDSM_ACC_MODULUS3	dig	DDSM_MSK	□	□	00	0 0 0 0 0 0 0 0
0128	DDSM_ACC_MODULUS4	dig	DDSM_MSK	□	□	00	0 0 0 0 0 0 0 0

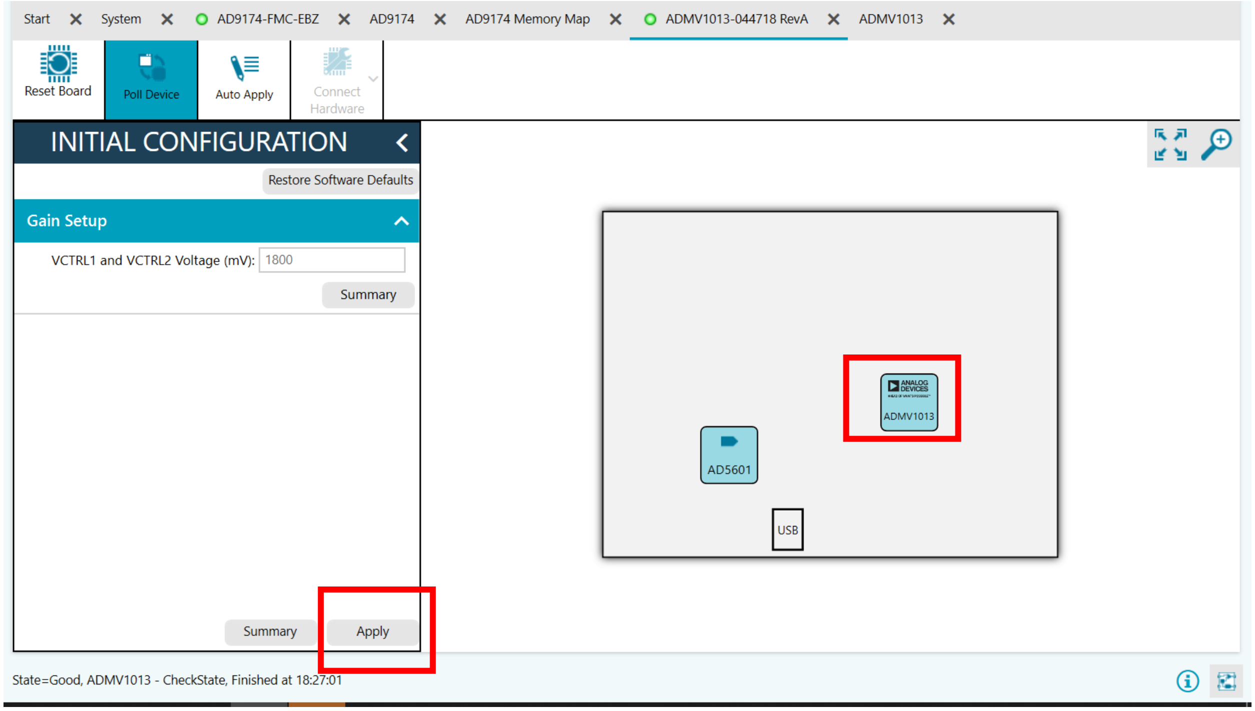
DDSM_PHASE_OFFSET1

If you don't set up the phase offset, you may still see a good constellation map from the spectrum analyzer as long as the DAC and spectrum analyzer are connected to the same reference frequency because the spectrum analyzer has its own algorithm to compensate for the phase offset. It's still recommended to set up the correct phase offset four our measurement.

Note: every time you click 'apply changes' the DAC will reconfigure the NCO phase offset and the spectrum analyzer will restart calculate the constellation map. It's very useful when you couldn't get a stable but 'rotating' constellation map from the spectrum analyzer due to algorithm issue.

3.2 ADMV 1013 Configuration

The configuration of ADMV 1013 is easy. Click apply to use the maximum gain. Then double click the icon of ADMV 1013.



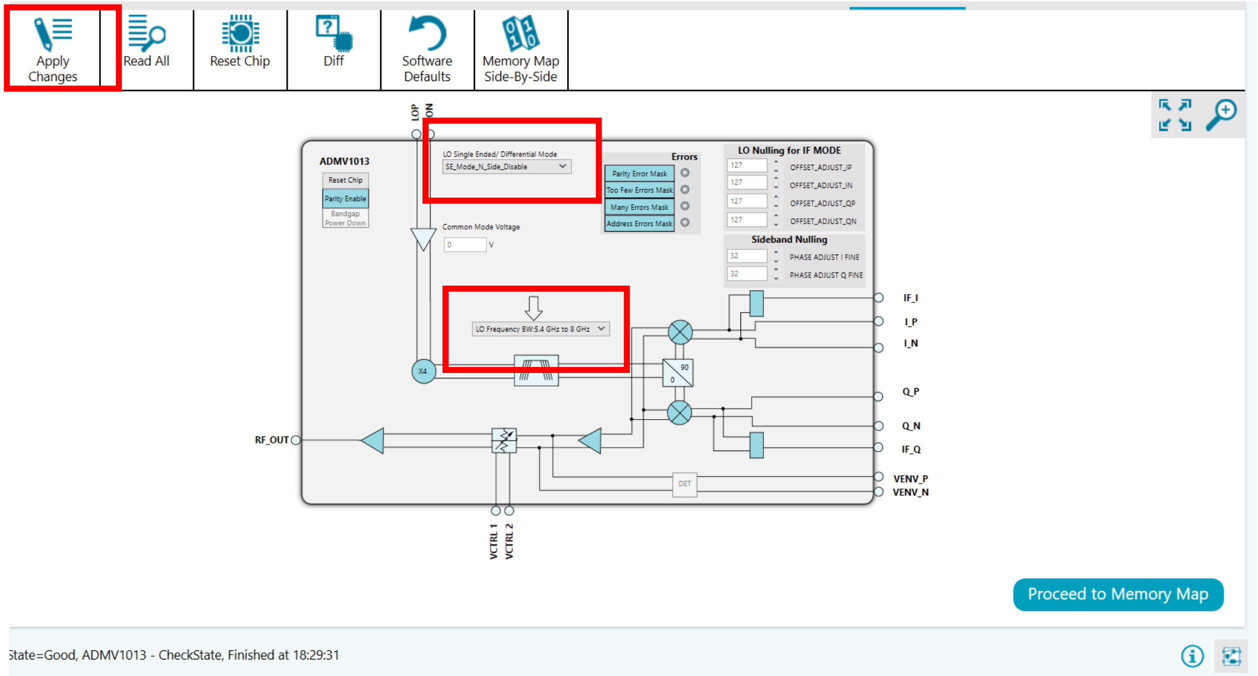
INITIAL CONFIGURATION

Gain Setup

VCTRL1 and VCTRL2 Voltage (mV): 1800

State=Good, ADMV1013 - CheckState, Finished at 18:27:01

In the new window, although not necessary, configure the bandpass filter for LO and LO port usage as follow the click apply changes (Here I'm using the LO_P as LO input port):



LO Single Ended/ Differential Mode: SE_Mode_N_Side_Disable

Common Mode Voltage: 0 V

LO Frequency BW:5.4 GHz to 8 GHz

LO Nulling for IF MODE

127	OFFSET_ADJUST_IP
127	OFFSET_ADJUST_IN
127	OFFSET_ADJUST_QP
127	OFFSET_ADJUST_QN

Sideband Nulling

32	PHASE ADJUST I FINE
32	PHASE ADJUST Q FINE

State=Good, ADMV1013 - CheckState, Finished at 18:29:31

You have done with the hardware & software set up!

4. Spectrum analyzer usage

To see the correct constellation map, you need to configure the spectrum analyzer using the right oversampling rate, symbol rate and RRC filter shape.

The maximum sampling frequency for the spectrum analyzer is only 128MHz above 20GHz, however the sampling frequency for the DAC is fixed to 368.64MHz. To solve this dilemma, we can use another function called 'QAM_circular_fin_slow_speed'. This function will repeat every symbol 3 times (you can also edit the code for lower speed) so that the actual sampling frequency would be 1/3 of the original (368.64MHz). In real measurement, this may not be necessary.

On the spectrum panel, push 'MODE' soft button, then click 'VSA' on the screen. Then push 'MEAS CONFIG' on the control panel. Then click 'Settings Overview'. Click 'Modulation Signal Description' to set up the correct modulation type (QAM 16/64) and symbol rate.

The symbol rate can be calculated as follow, assuming using a 3x slower DAC clock and an oversampling rate of 32:

Symbol Rate=368.64/(32*3)=3.84MHz.

Set up the Transmitter Filter as 'RRC', 'Alpha/BT' as the value you set in the sequence generation code (0.8 for my code).

Then you finish the set up for 'Modulation Signal Description'.

Next, click 'Frontend & I/Q Capture'. Set the frequency correctly (4*flo+1G/4*flo-1G). For the attenuation setting, sometimes moderate attenuation will lead to better constellation measurement but bear in mind any attenuation will degrade the EVM. Then click 'I/Q Capture'. Set the 'Capture Ov (Sample Rate)' as the oversampling rate specified in the code. The value in my code is 32. Higher oversampling rate would lead to narrow bandwidth, better constellation map and fewer symbols since the memory length is fixed.

The spectrum analyzer set up is now finished.

You should be able to see a good constellation map now especially for 16 QAM. For 64 QAM, the constellation may not be quite 'square-like' due to the algorithm issues. But it's fine as long as it's static. I hope you can find a way to fix it.