

Synchronizing the Giga-Sample ADCs Interfaced With Multiple FPGAs

Purnachandar Poshala, Pavan Shetty

High Speed Data Converters

ABSTRACT

Phase matching between the receiver channels are required in Phase Array Receiver. These receivers are typically used for radar, communications and target tracking systems. The receiver channels contain high-speed giga-sample ADCs for RF sampling and are generally interfaced with FPGAs for capturing the ADC data. The key components in the receiver are giga-sampling ADCs. Phase matching of these ADCs across receiver channels is required for phase-array receivers. When multiple receiver channels are required in the system, sometimes it is possible that different ADCs are connected to different FPGAs. The phase matching of the overall receiver depends on the matching of the ADC samples captured by different FPGAs.

This application note describes how such phase-matching requirements can be met using TI's giga-sample ADCs ADC12D1xxx family of parts utilizing time stamp, autosync and aperture delay correction features.

Contents

1	Introduction	2
1.1	Why is Phase Matching Important Between the ADCs?.....	2
1.2	Typical Receiver Signal Chain Blocks Indicating Phase Variations	3
1.3	Reasons for Phase Difference Between Multiple ADCs	5
2	Correcting the Phase Difference Between Multiple ADCs.....	7
2.1	AutoSync	7
2.2	Timestamp.....	8
2.3	Aperture Delay Adjustment Feature.....	11
3	System Implementation - Testing With ADC12D1600RF Evaluation Modules for Correcting the Phase Difference Between Multiple ADCs.....	11
4	Test Results	15
5	Conclusion	16
6	References.....	16

Figures

Figure 1.	Phase-Array Radar Receiver System	3
Figure 2.	Receiver Signal Chain With Phase Error Elements	4
Figure 3.	Sample Block Diagram for a Typical System With 4 ADCs in DES, DEMUX Mode	5
Figure 4.	Multiple ADC Synchronization Test Setup.....	13

Tables

Table 1.	Raw Data Stream With Sample Errors.....	9
Table 2.	Corrected Values (Using Timestamp Datastream)	9
Table 3.	Phase Variation Between Parallel Signal Paths on Separate ADCs.....	15
Table 4.	Phase Variation of Signal Chain After Quick Power Reset of System	16

1 Introduction

1.1 Why is Phase Matching Important between the ADCs?

A number of applications rely on uniformity of the input to output delays of multiple signal channels for accurate measurements. An example of such an application is the phased-array radar system. The received signals from different receivers are combined coherently for beam forming for searching, tracking or search, and track simultaneously in phased-array radars.

Active phase-array receivers have coupled transmitters and receivers on the same block on each array element while passive phase-array receivers receive the same RF signal input from a remote source. The beam steering on a transmitter is electronically accomplished by incrementally phase shifting the signals to the antennas of the array. The receiver system consists of a linear array of individual antenna elements. On the receiving end, using phased array antennas allows for spatial filtering to scan the reflection from a particular angle. The phase delay elements act to cancel out the time delay between the outputs of the array elements on receiving an input RF signal at an angle such that the signals received from the other angles are suppressed when summed up.

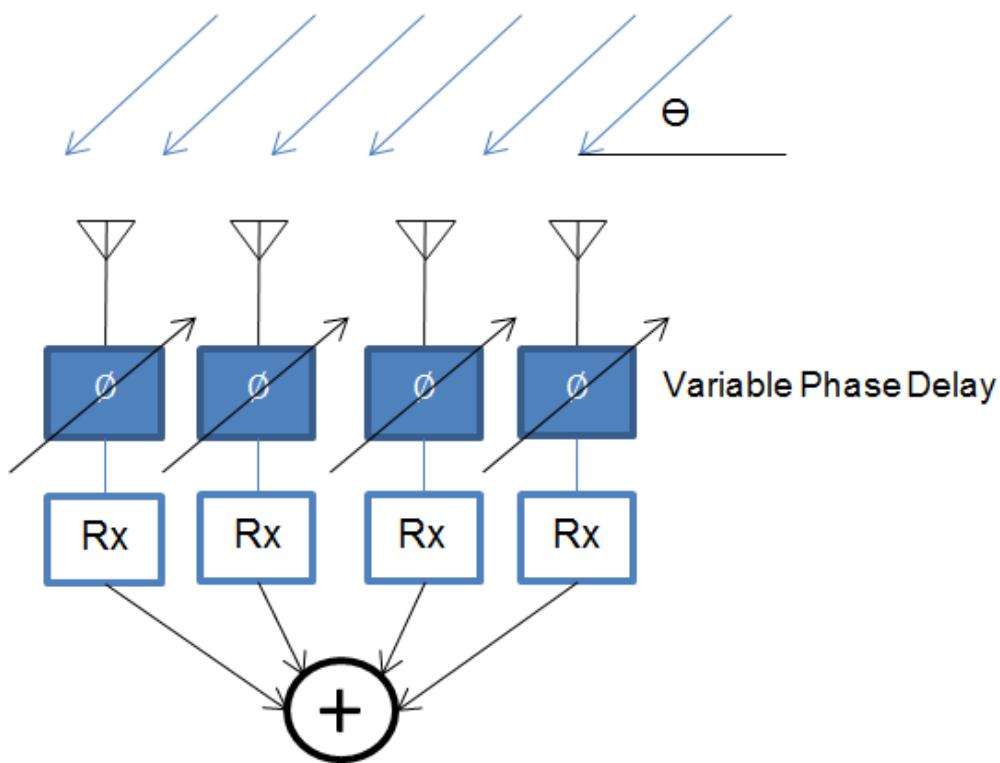


Figure 1. Phase-Array Radar Receiver System

Usage of wideband ADCs like TI's giga-sample ADCs allow elimination of a down-conversion stage by sampling higher frequency bandwidths of the signal. Such systems may need only one down-conversion stage to convert from the K band, C band, or X-band to L-band frequencies. The L-band frequencies can be directly sampled using giga-sampling ADCs. These giga sample ADCs have 3-dB bandwidth of 2.7 GHz.

The phased-array receivers determine the distance and location of the object by estimating the phase information seen on the receiver array. Any phase-difference variation induced in the ADC channels or the receiver signal chain will cause an error in this estimation. This application note refers to a methodology which allows us to compensate for the errors induced due to variations on ADC in the receiver signal chain while taking care of small variations in the signal path (if phase variation in the time domain is lesser than one sample period).

1.2 Typical Receiver Signal Chain Blocks Indicating Phase Variations

[Figure 2](#) illustrates the receiver signal chain in a conventional system with giga-sample ADCs and the possible causes of phase variations.

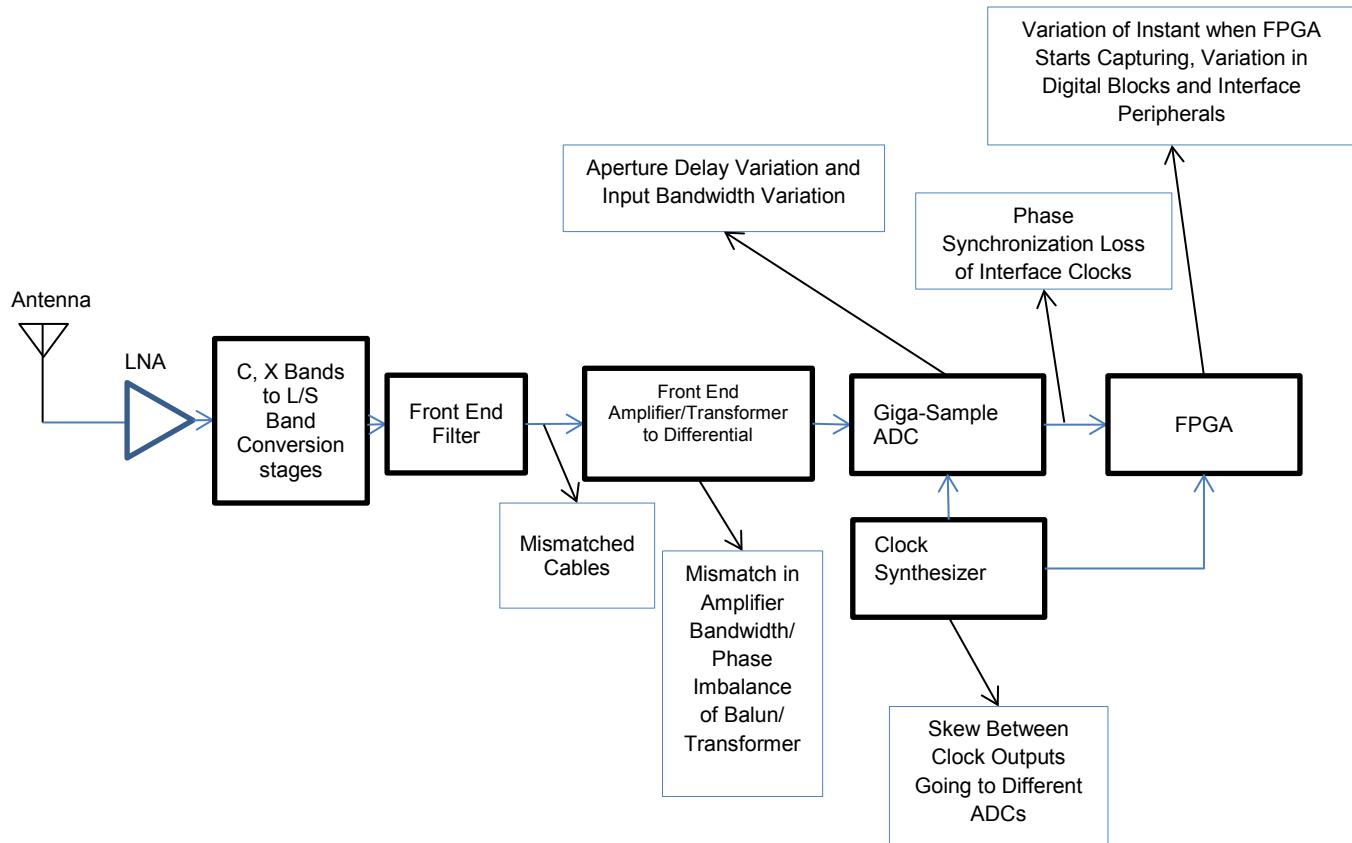


Figure 2. Receiver Signal Chain With Phase Error Elements

The designs containing multiple receiver chains might include multiple ADCs connected to one or more FPGAs. It uses multiple giga-sample ADCs (for example, ADC12D1600RF) whose outputs are captured by single or multiple FPGAs using the LVDS outputs and clock generated from the ADC. The ADCs sample the input on clocks generated from a common source. We will consider ADC12D1600RF in our discussion throughout this application note even though it is applicable for all the ADCs in the ADC12D1x00RF family of parts like ADC12D500RF, ADC12D800RF and ADC12D1800RF, and so forth. Refer to the ADC datasheets for more information on these ADC features. ADC12D1600RF may be used in dual-channel non-DES (Dual Edge Sampling) mode or single channel interleaved DES mode for higher sampling rates. In DES mode the two ADCs inside one chip are interleaved (sampled on 180° phase shifted clocks) where one ADC samples the input on rising edge and the other ADC samples the same input on the falling edge. In non-DES mode, the two ADCs inside the chip sample the input signal at the same clock edge and can be treated as two separate channels. [Figure 3](#) has the system block diagram for a typical use case.

The output streams are de-multiplexed to multiple output buses to allow for lower transfer clock speed, which can be supported by the FPGA. The DCLKs are generated by the ADC and provided to the FPGA to latch in the de-multiplexed data sent by the ADC. These are generated by dividing down the sampling clock. The phase of the divided down clocks are synchronized with the auto-sync feature which is described in the [Correcting the Phase Difference Between Multiple ADCs](#) section.

The FPGAs in the system start receiving data from multiple ADC output lines after receiving an external trigger from the master controller which may be another FPGA or a processor. The external trigger to the FPGA should have minimum skew between them, which can be corrected by the timestamp feature which is explained in the [System Implementation - Testing With ADC12D1600RF Evaluation Modules for Correcting the Phase Difference Between Multiple ADCs](#) section.

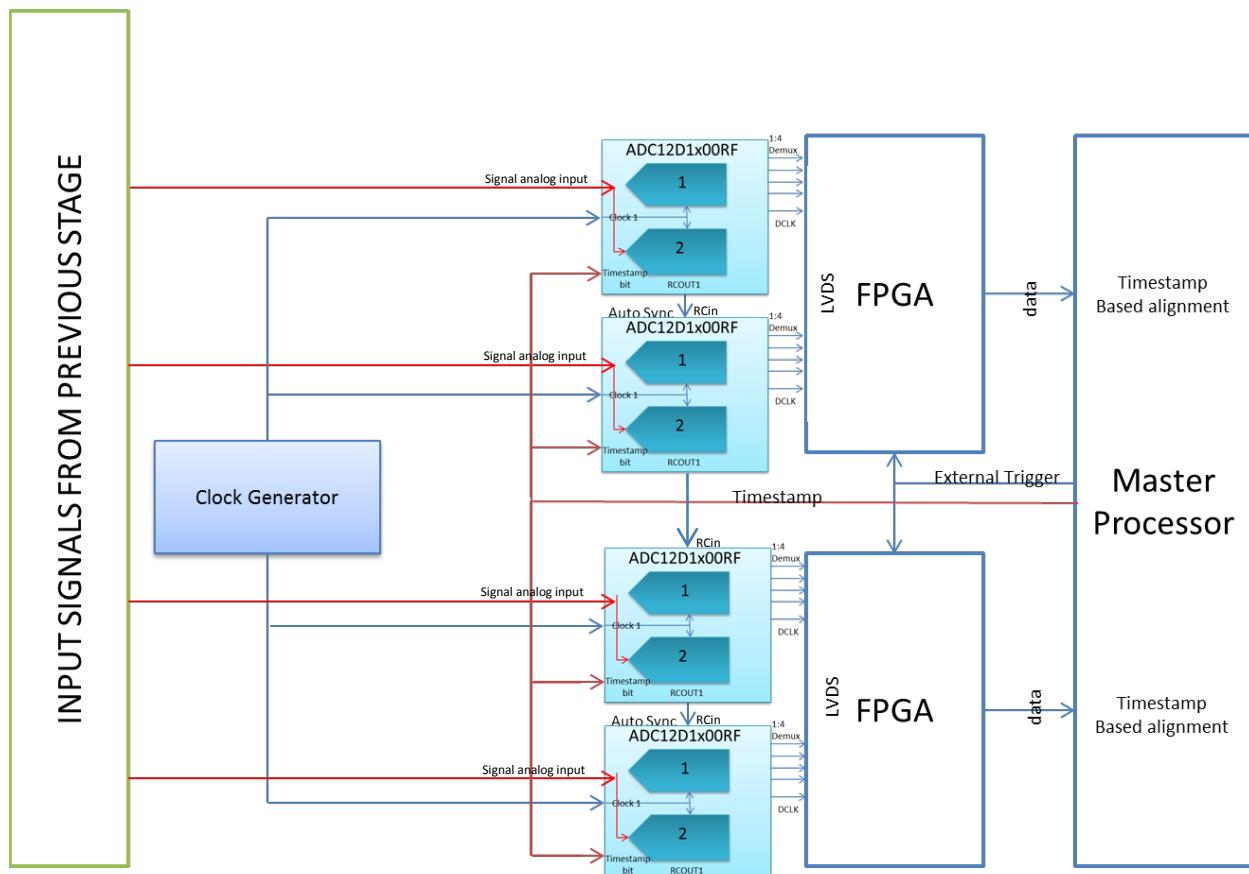


Figure 3. Sample Block Diagram for a Typical System With 4 ADCs in DES, DEMUX Mode

1.3 Reasons for Phase Difference Between Multiple ADCs

Phase difference between the ADCs may arise due to:

- **Loss of synchronization between different ADC-FPGA interface data clocks (DCLK):** To reduce the data rate from the ADC to FPGA, the ADC12D1x00 sends the data output on multiple data buses to transmit data at a rate supported by the presently available FPGAs. For example, it sends the 12-bit data output on 4 data buses (48 data lines for 12×4 bits) and the data rate is reduced by 4 in a particular Demux mode. The ADC input sampling clock is divided down accordingly to generate the output data clock (DCLK). However, the phase of the generated clock is not fixed by design. For example when used in a 4:1 Demux DDR mode, the DCLK is divided by 4 and can assume 0, 90, 180, or 270 degree phase for the DCLK generation. When multiple ADCs are used in the design for the multiple receive channels, the DCLK for each of the ADC output may or may not be phase synchronized.

This variation may lead to samples stored at different index locations inside the FPGA even though the input is sampled at the same sampling edge by different ADCs.

- **Variation in input stage of ADCs:** Phase difference variation in baluns, splitters and amplifiers in the input signal path may cause phase difference between two identical ADC signal chains.
- **Skew between the sampling clock generator's outputs:** The sampling clocks generated by the clock synthesizer may have a phase variation due to skew between the clock buffer outputs of the clock synthesizer system which translates into phase variation between the ADC channels. Also unmatched traces or cables used for clocks may create the phase variation.
- **Variation in the aperture delay over different ADC devices:** Aperture delay is defined as the amount of delay, measured from the sampling edge of the CLK input, after which the signal present at the input pin is sampled inside the device. It actually arises due to delay in the clock distribution tree inside the chip which results in the clock edge reaching the sample and hold after the analog input reaches the device. Generally, the aperture delay value provided in the datasheet is by design and can vary from device to device. More details on ADC Aperture Delays can be found in the *From Sample Instant to Data Output: Understanding Latency in the GSPS ADC* application note ([SNA198](#)).
- **Variation in analog input bandwidth of ADCs:** The input bandwidth of the ADCs may vary due to silicon variations and cause phase difference variation between multiple ADCs. This bandwidth can also change with temperature.

Assuming a first order low pass filter model for the ADC sampling circuit:

Transfer function:

$$H(f) = \frac{1}{1+j\frac{f}{F}} \quad F: \text{Analog 3 dB bandwidth of ADC} \quad (1)$$

Input to output phase difference:

$$\theta(f) = -\tan^{-1} \frac{f}{F} \quad (2)$$

Assume total phase variation between 2 ADCs is $\Delta\theta$,

Variation in phase difference:

$$\Delta\theta(f, \Delta F) = -[\tan^{-1}(\frac{f}{F+\Delta F}) - \tan^{-1}(\frac{f}{F})] \quad (3)$$

For example, for ADCs with 3-dB bandwidth of 2.7 GHz, with input frequency at 1500 MHz, and with a bandwidth variation of 5%, the phase variation between multiple ADCs is around 1.2 degrees.

This clearly indicates that the ADC Input bandwidth variation can cause input frequency dependent phase variation between two ADCs.

- **Delay between the data capturing inside FPGAs:** Different FPGAs may capture a dataset starting at different instants due to: (1) variation in external trigger received to start data capture or (2) unsynchronized clocks supplied to the two FPGAs due to which the FPGAs receive the external trigger at different instants.

Overall, from the previous discussion, the phase difference between two ADCs occurs in the following forms:

1. Misalignment of samples: Two separate outputs may be off by multiple samples and result in a phase difference. Thus, the time difference between two readings is a multiple of the sampling period.
2. Phase difference less than one sample period: The delay between the two channels may vary by a value lesser than the sampling period which is not due to sample misalignment but variations in the analog input path or variations in the clock path.

The numeric value of phase difference between two channels while testing with a single tone input may not reflect the former case (case 1) correctly since the time difference may represent a phase variation greater than 360° which will roll back. For example, if a 1.9-GHz waveform sampled at 3600 Msps by two different ADCs is misaligned by 3 samples, the resultant time difference is 0.83 ns, which is 567 degrees. However, when two such single tone signals are compared and the phase difference is computed, the phase difference appears to be 207 degrees (567 degrees = $360 + 207$ degrees, roll backs to 207 degrees). These 207 degrees actually translate to 0.303 ns which is different from the actual phase delay of 0.83 ns and appears to be a subsample period delay (case 2). Therefore, multiple sample variation should be resolved before subsample period variation to correct the total phase variation. This is accomplished with the [Timestamp](#) feature discussed in section [2.2](#).

2 Correcting the Phase Difference Between Multiple ADCs

The key features of the ADC used for correction of the Phase Difference between the multiple ADCs follow:

- AutoSync
- Timestamp
- Aperture delay correction

2.1 AutoSync

The AutoSync feature on the ADC12D1600RF can be used to synchronize the divided down DCLKs from multiple ADCs such that the data is latched at the same time inside FPGA. For a detailed description on the autosync feature refer to *AN-2132 Synchronizing Multiple GSPS ADCs in a System: The AutoSync application note ([SNA073F](#))*.

The basic steps for implementing the AutoSync feature are as follows:

- Configure one ADC as the master and the rest of the ADCs in the systems as slaves and drive the RCIN of the slaves with RCOUTs derived from the master ADC. Alternatively, RCIN can be driven externally.
- Set up DRC bits of the ‘Eh’ register of the ADC to place DCLK in a stable region with respect to setup and hold times.
- Configure the SP (select phase) bits of the ‘Eh’ register such that the DCLKs of different ADCs are in phase when seen in an oscilloscope.

- When designing the board with these ADCs, it is advised to have a DCLK monitoring tap for each ADC to view it on an oscilloscope for AutoSync feature implementation. Before monitoring the DCLKs in oscilloscope, it is advisable to find the phase between two channels of the high bandwidth oscilloscope used for correctness of the oscilloscope channels phase matching. It is possible that 2 channels of the oscilloscope have an internal phase difference between them due to sampling at different edges of the clock inside the oscilloscope. The same can also be verified by inverting the inputs and observing the change in the phase difference.

The AutoSync is the first step for correcting the phase difference between multiple ADCs before implementing the other features, like timestamp and aperture delay for correcting the phase difference between multiple ADCs.

2.2 Timestamp

Timestamp is an excellent feature provided in the ADC12D1x00RF series ADCs allowing input of a synchronizing signal to the DCLK_RST pin (called as D0 timestamp pin) of the ADC which is converted into a 1-bit value on the ADC output at the D0 bit. The signal faces the same delay as faced by the input signal through the ADC channel pipeline, that is, $t_{\text{latency}} + t_{\text{output delay}} - t_{\text{aperture delay}}$ along the signal path. Therefore, any variation in the internal delay of the ADC or variation in the instant at which the FPGA starts capturing the ADC readings can be identified by observing the D0 bit (timestamp bit) of the output for multiple ADCs. Once identified, this can easily be digitally corrected inside the FPGA by matching the samples with respect to the timestamp bit transition to a HIGH or LOW state. Ideally, if there are no variations in the input signal path and no other variations, the D0 bit (timestamp bit) should be latched inside the FPGA for different ADCs at the same time.

As the ADC's LSB bit (D0 bit) is used to represent the timestamp signal, and since the ENOB of the ADC does not exceed 10 bits, usage of this feature doesn't result in any significant loss in SNR of the ADC. It can be used to correct any sample delays between two ADC output streams because a matched timestamp transition on the ADCs represents input samples received on the same clock edge.

An identical timestamp signal is sent to the multiple ADC modules in the system. The output data streams can be aligned according to the timestamp signal edges since it is known that the ADCs are receiving the input signals at the same time. However, care should be taken to reduce mismatches in the system and maintaining same phase for output clocks by autosync before timestamp based alignment to reduce the misalignment that is required to be corrected by timestamp.

If the same timestamp signal is sent to two separate ADCs (with minimum fanout skew), ideally the ADC signal would show the same LSB variation on the output bit streams. However, the skew errors discussed previously can lead to phase delays between the two paths which could be greater than one sampling period, thereby showing variation in the LSB bitstreams. The instant at which the LSB changes from HIGH to LOW (that is, the reading value changes from odd to even) or vice versa may vary between the parallel signal streams. For example, consider a case where the ADC1 samples set LSB is changing to HIGH at sample no. 45 and ADC2 sample set LSB is changing to HIGH at sample no. 48. In this case, we adjust the sample no. 48 of ADC2 to sample no. 45 of ADC1 as same sample instant by moving up the samples of ADC2 by 3 locations in the data-stream (moving the 45th, 46th, and 47th sample of ADC2 to the 42nd, 43rd, and 44th location and keeping the 48th sample of ADC2 at the location of ADC1 45th sample). In this way, the 48th sample of ADC2 is adjusted to a similar location of 45th sample of ADC1.

After this change, all future samples of ADC1 and ADC2 should change the LSB bits to HIGH at the same sample instant. This has to be checked for at least the next few LSB bit changes for verification of the phase-matching implementation in the actual system.

So the raw output when the same analog input signal is passed on to the parallel signal chains could be as shown in [Table 1](#) (12-bit output with the same timestamp signal fanned out to the two ADCs).

Table 1. Raw Data Stream With Sample Errors

ADC 1 Index	ADC1 Reading	ADC 2 Index	ADC2 Reading
40	2358	40	2110
41	1718	41	2152
42	2304	42	1842
43	1894	43	2358
44	2042	44	1736
45	2207	45	2316
46	1777	46	1896
47	2383	47	2040
48	1743	48	2213
49	2293	49	1793
50	1983	50	2375
51	1971	51	1745
52	2277	52	2291
53	1733	53	1995

After correction (by offsetting initial data points), the data stream is as seen in [Table 2](#).

Table 2. Corrected Values (Using Timestamp Datastream)

ADC 1 Index	ADC1 Reading	ADC 2 Index	ADC2 Reading
40	2358	43	2358
41	1718	44	1736
42	2304	45	2316

43	1894	46	1896
44	2042	47	2040
45	2207	48	2213
46	1777	49	1793
47	2383	50	2375
48	1743	51	1745
49	2293	52	2291
50	1983	53	1995
51	1971	54	1973
52	2277	55	2273
53	1733	56	1747

Timestamp can be used to correct multi-cycle errors when the frequency is less than that of the sampling clock.

2.2.1 *Timestamp Implementation Procedure*

- The timestamp signal can be a trigger generated by the FPGA or can be generated as a low-speed periodic signal by the clocking device in the system. Make sure the ADCs receive the timestamp signal at the same instant.
- The time period of the timestamp signal to correct errors due to the variations in ADC internal delay or variation in time instant at which the FPGA captures ADC output data should be greater than the error caused by any of the these factors, to avoid repetitions of the timestamp signal edges within an error region. Such a case will create an ambiguity as to which edges of the timestamp signal should be aligned. For example, if the maximum error between the instant when two FPGAs capture ADC data is 10 ns then the frequency of the timestamp signal should be lesser than 100 MHz to have the period of the timestamp signal more than 10 ns.
- Although the timestamp signal need not be synchronous with the sampling clock, it is beneficial to have the timestamp signal edge in the middle of the sampling clock period which is a stable region to counter small variations in the timestamp signal which go to the ADCs. This variation can arise from phase difference in the timestamp signals applied to multiple ADCs in the system or due to phase difference in the sampling clocks supplied to the multiple ADCs. This might result in the timestamp signal edge occurring at two different edges of the sampling clock across multiple ADCs which will lead a cycle delay on the output even after timestamp alignment. To correct this error, the timestamp signal can be synchronized to the sampling clock with a fixed delay to ensure that the timestamp edges occur in the middle of the sampling clock levels.
- The outputs of the two different ADCs after capturing inside the FPGA can be aligned by matching the nearest timestamp signal edges in the data array, such that odd and even outputs values are next to each other. As timestamp bit is the D0 bit, the HIGH on the DCLK_RST (time stamp pin) pin makes the data as odd numbers and LOW time on the DCLK_RST makes the data as even numbers.
- Using a higher frequency timestamp signal can cause mixing spurs with the input to the ADC due to capacitive coupling with signal path inside the ADC. A low frequency pulse is

recommended with time period greater than maximum phase error possible in system, but lesser than half the sample stream width.

2.3 Aperture Delay Adjustment Feature

The timestamp method previously described can be effectively used for correction of multi-sample delay errors. For correcting the phase errors less than one sample period, the ADC aperture delay adjust option can be used. The aperture delay adjustment is a programmable delay which can be added to the sampling clock before the sample is acquired. This delay can be adjusted using the coarse (approximately 340 fs steps) or finer (approximately 34 fs steps) adjustment options provided in the ADC12D1600RF.

2.3.1 Aperture Delay Implementation Procedure

Implement the aperture delay using the following steps:

1. Enable STA bit (aperture delay adjustment bit) of register ‘Ch’ on all ADCs in the system, even if the clock for a particular ADC is not to be delayed, because enabling the bit adds a definite delay to the sampling clock and should be equalized for all ADCs before phase matching.
2. Use a test input signal and minimize the phase difference between the ADC outputs by changing the course and fine adjust magnitudes. However, PVT variations can be expected on the delay generated (± 95 ps for 825-ps delay with course adjust and ± 300 fs for 2.3-ps delay with course adjust).

Varying aperture delay may affect signal integrity, especially at higher sampling rates and thus a minimal amount of adjustment should be used. The use of this feature should be avoided at maximum sampling rates of the ADC. Please check the corresponding ADC datasheet for the maximum limit on this feature. Adding additional delay in the clock path adds to the aperture jitter and delays all clocks generated by the clock tree inside the ADCs. Care should be taken to make sure the delay added by the aperture adjust (maximum of 825 ps) never approaches the sampling period (maximum of 625 ps). The adjustment delay exhibits a variation of approximately 10% with PVT variations. Hence, for finer phase matching the adjustment should be calibrated for changes in ambient conditions due to events like a delayed power on. The adjustment should be kept at a minimum to reduce PVT variations.

3 System Implementation - Testing With ADC12D1600RF Evaluation Modules for Correcting the Phase Difference Between Multiple ADCs

For synchronizing the two different giga-sampling ADCs connected to two different FPGAs, the ADC12D1600RFRB EVMs were selected where each EVM (Evaluation Module) contains the ADC12D1600RF ADC and Xilinx Virtex 4 FPGA for data capture. A PC or laptop is used for capturing these two ADC data captured inside two different FPGAs by using the Wave Vision 5 software.

The test setup is made for testing the ADC EVMs and synchronizing the two ADCs by implementing the autosync, timestamp and aperture delay corrections for correcting the phase difference between the multiple ADCs. Even though ADC12D1600RFRB EVM is used here, the same implementation is applicable for other ADCs in this family like ADC12D500RF/800RF/1000RF/1800RFRB EVMs within their sampling rate limitation.

Use the following steps for the test setup:

1. A 1474.56-MHz LVPECL clock is generated with an LMK04808 EVM. The ADC12D1600RF is used in DES mode, therefore, the effective sampling rate of the ADC is 2949.12 Msps. The DES mode is selected using the *Wave Vision 5* GUI software used for capturing the ADC data. This sampling rate is selected as the LMK04808 EVM can generate this easily using the on-board 122.88-MHz OCXO. Other frequencies are possible but again the OCXO needs to be changed accordingly. To simplify, the existing OCXO is used.
2. The 1475.56-MHz clock is LVPECL output from the LMK04808. But it is fed as single ended due to the fact that the ADC EVM can take only single-ended input and internally on the EVM it converts the clock to differential ended before feeding it to the ADC. This clock chip can generate clocks with a maximum clock skew of about 30 ps.
3. A timestamp signal of 20-MHz LVCMS, delayed by 850 ps generated with the same LMK04808 EVM. The maximum skew is of 100 ps between this LVCMS timestamp signal output and LVPECL ADC Sampling clock outputs as per LMK04808 datasheet ([SNAS489](#)). The delay of 850 ps is generated to make sure that the timestamp signal is in the middle of the sampling clock. This number can vary based on the setup and based on the clocking chip and the way of generating the clocks for timestamp and sampling clock.
4. Input frequencies are generated from signal generator in the range of 1700- to 2700-MHz band which can be sampled directly by the ADC. As the ADC input bandwidth is a maximum of 2700 MHz (3 dB BW), this range was selected. Also to check the phase variations for wide frequency band, the frequency band of around 1 GHz is selected.
5. All the SMA cables in the setup were length-matched, however small mismatches could be expected due to the same. In the test setup, both ADCs were fed with the same single-tone signal through a splitter and the signal was converted into differential using Balun boards.
6. The external trigger feature is enabled from the *Wave Vision 5* software which is available for capturing the data from multiple ADC12D1600RF EVMs at the same time. Signal generator output is connected to two EVM external trigger connections using the splitter. Using this feature, ideally the two FPGAs on two EVMs will capture the ADC samples at the same instance but which might not be the case practically which will be corrected.
7. The ADC die temperature measure on the EVM is approximately 83°C using the *Wave Vision 5* software.

Testing Methodology:

1. [Figure 4](#) shows the test setup.
2. For testing the autosync feature, both the DCLKs from both the ADCs are connected to oscilloscope and monitored the phase difference.
3. Before starting the actual test for autosync implementation, make sure that both channels of the oscilloscope are phase matched. Please use the de-skew feature of the oscilloscope for nullifying any phase difference of the oscilloscope itself.

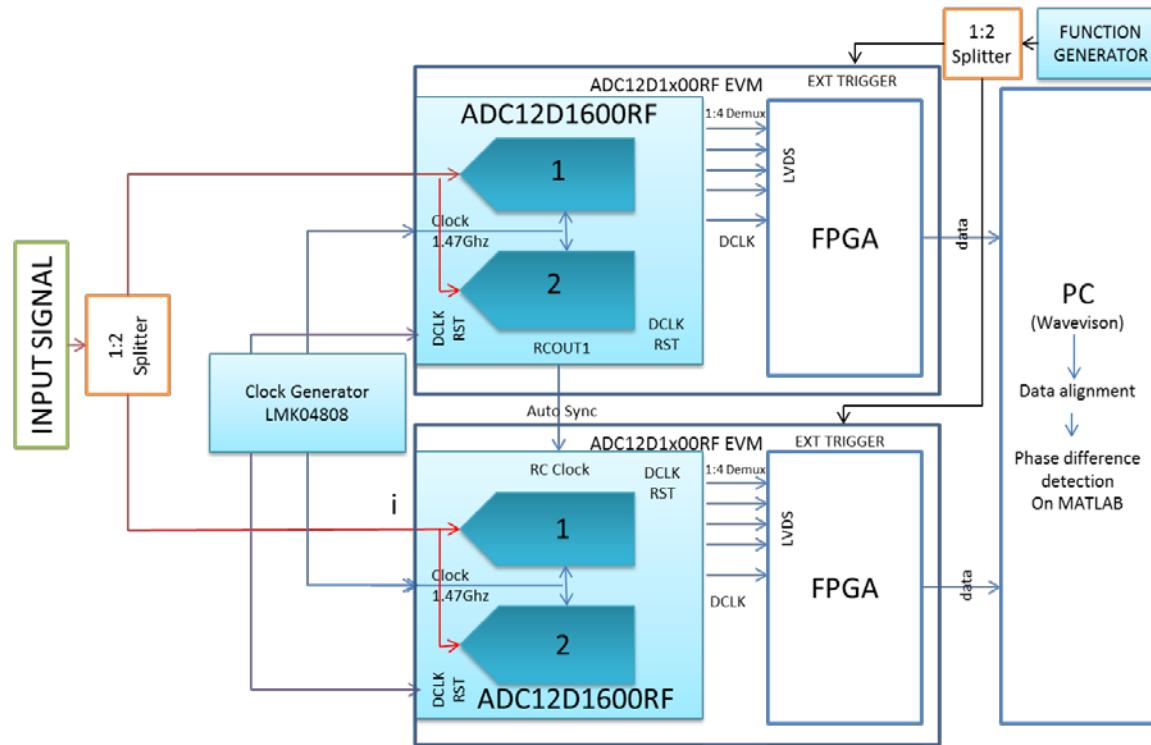


Figure 4. Multiple ADC Synchronization Test Setup

4. If the phase difference between the two ADC DCLKs monitored on the oscilloscope exists, this can be corrected by changing the DRC delay bits, SP (Select Phase) bits using Wave Vision 5 GUI.
5. Make sure that the DCLKs are in a stable region by verifying the entire range for DRC delay and finalizing the value based on the stable DCLKs. This way, the 90, 180, and 270 degree phase variation between the ADCs can be adjusted using the autosync feature.
6. After this, there might be still some phase difference between the two DCLK outputs due to the variation in the aperture delay in the ADC or skew in the clock source or variation in the input signal path. This variation was adjusted with aperture delay variation previously discussed.
7. Around 200 ps variation was observed between the DCLKs while testing and this was corrected using the aperture delay variation. However, the phase difference between test inputs was nullified by adding an additional delay of 50 ps to the slave as opposed to the required 200 ps based on the test data observations. Therefore, it is necessary to experimentally calibrate the aperture delay for a particular test input on the bench.
8. The clocks for the FPGAs on the EVMs are not synchronized with the input and independently derived on the EVMs. Hence, the instant at which the FPGA can capture the external trigger can vary at most by the clock period of this clock on the EVM which is 10 ns (100-MHz clock). The output arrays of the two ADCs can be off by 0 to 10 ns which is reflected as multiple samples variation and is corrected by timestamp alignment in this setup.

9. The samples captured on Wave Vision 5 software on two separate systems was brought on an excel sheet and the process of timestamp alignment was carried out. The absolute phase of signal at the frequency with maximum amplitude was found by carrying out a fast-Fourier transform (FFT) in MATLAB®. The difference of phase between the two signals was used to calibrate the aperture delay adjust register of the ADC.
10. The values for aperture delay, timestamp alignment, and autosync will vary system to system. While autosync does not need further calibration (for example, after a power on) due to the facility of DRC delay, aperture delay should be calibrated for different ambient conditions to for better phase matching. Actual system design can have a provision for calibrating for different temperature test conditions.

4 Test Results

Table 3 shows the phase difference variation for frequencies in the 1700- to 2700-MHz band. Multiple readings were taken but only 4 readings of phase difference variation are shown in **Table 3**. Also, **Table 4** is shown after power off and power on condition. In this setup with the ADC EVMs, the variation in the phase difference between the two ADC channels is observed to be **less than 4 degrees** across the wide frequency band of 1000 MHz.

Table 3. Phase Variation Between Parallel Signal Paths on Separate ADCs

Input Frequency	Phase Difference Between ADC1 and ADC2 Reading 1	Phase Difference Between ADC1 and ADC2 Reading 2	Phase Difference Between ADC1 and ADC2 Reading 3	Phase Difference Between ADC1 and ADC2 Reading 4	Maximum Phase Difference Variation Between ADC1 and ADC2	Mean Value
MHz	Degrees	Degrees	Degrees	Degrees	Degrees	Degrees
1700	0.0893	-0.021	-0.0423	-0.1092	0.1985	-0.0208
1750	0.7491	0.7671	0.6026	0.7085	0.1645	0.706825
1800	1.8932	1.8165	1.9163	1.8211	0.0998	1.861775
1850	2.3026	2.2619	2.2583	2.2263	0.0763	2.262275
1900	3.3583	3.3585	3.3172	3.1734	0.1851	3.30185
1950	3.613	3.4873	3.5006	3.5427	0.1257	3.5359
2000	3.7045	3.5773	3.5386	3.4483	0.2562	3.567175
2050	2.9455	2.7966	2.7638	2.8982	0.1817	2.851025
2100	1.7913	1.6877	1.6292	1.8926	0.2634	1.7502
2150	1.6602	1.6737	1.6692	1.5917	0.082	1.6487
2200	2.218	2.3801	2.3052	2.3232	0.1621	2.306625
2250	1.3906	1.3221	1.3906	1.2285	0.1621	1.33295
2300	1.43	1.4234	1.3718	1.4589	0.0871	1.421025
2350	2.9843	3.1667	3.3356	3.2956	0.3513	3.19555
2400	3.4489	3.4196	3.4667	3.3111	0.1556	3.411575
2450	2.6006	2.5437	2.6658	2.5193	0.1465	2.58235
2500	1.1247	1.3544	1.4902	1.3777	0.3655	1.33675
2550	0.0092	0.025	-0.2102	-0.156	0.2352	-0.083
2600	-0.4271	-0.3281	-0.4078	-0.4131	0.099	-0.39402
2650	-0.4862	-0.3742	-0.3588	-0.356	0.1302	-0.3938
2700	1.8366	1.7826	1.9347	2.0068	0.2242	1.890175
Overall variation						3.9612

Table 4. Phase Variation of Signal Chain After Quick Power Reset of System

Input Frequency	R1	R2	Variation	Mean
MHz	Degrees	Degrees	Degrees	Degrees
1700	-0.4202	-0.4454	0.0252	-0.4328
2700	2.3081	2.0571	0.251	2.1826

5 Conclusion

The implementation for correcting the phase difference between the Giga-Sample ADCs like ADC12D1x00RF was provided, when used in different receiver channels connected to different FPGAs using the ADC Evaluation Modules test setup and corresponding test results were provided in this application note. The features of the ADC like autosync, timestamp and aperture delay adjustment were used for implementing the same.

6 References

1. Plisch, M. (2013). *From Sample Instant to Data Output: Understanding Latency in the GSPS ADC*. ([SNAA198](#)).
2. Texas Instruments. (2011, rev 2013). *AN-2132 Synchronizing Multiple GSPS ADCs in a System: The AutoSync* ([SNAA073](#)).
3. *ADC12D1600RF datasheet* ([SNAS519](#))

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as "components") are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI's terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers' products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers' products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI's goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or "enhanced plastic" are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have **not** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products	Applications		
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com	TI E2E Community	
OMAP Applications Processors	www.ti.com/omap	e2e.ti.com	
Wireless Connectivity	www.ti.com/wirelessconnectivity		