Method for Reducing Jitter in Multi-Gigahertz ATE

D.C. Keezer[1], D. Minier[2], P. Ducharme[2]
1- Georgia Institute of Technology, Atlanta, Georgia USA
2 – IBM, Bromont, Canada

Abstract

Controlling jitter on a picosecond (or smaller) time scale has become one of the most difficult challenges for testing multi-gigahertz systems. In this paper we present a novel method for reducing jitter in timing-critical ATE signals. This method uses a real-time averaging approach to combine multiple ATE signals and produces timing references with significantly lower random jitter. For example, we ***demonstrate a 3x reduction in jitter*** *by combining eight ATE signals (each with σ =4ps) to produce a low-jitter signal (σ =1.3ps). The measured jitter reduction is shown to closely match that predicted by theory. This counter-intuitive (but welcome) result is of general interest for the design of any low-jitter system, and is particularly helpful for multi-GHz ATE where precise timing is so critical.*

1. Introduction, Background, Motivation

At moderate frequencies (~100 MHz) picosecond jitter in ATE has tended to be a relatively minor concern, often overshadowed by other sources of timing errors (which have been on the nanosecond time scale). However, in multi-gigahertz ATE, great care is taken to reduce all timing error sources to the picosecond scale. With enough effort (including extensive calibration methods), today's ATE systems can all but eliminate most of the traditional sources of timing errors.

However, an exception to this has been the stubborn presence of jitter[1-3]. In theory, and (especially) in practice, jitter is always present to some degree and cannot be entirely eliminated. It is generally assumed that jitter can never be improved (i.e. that it only increases as new features are added to the system). Usually this does tend to be a valid assumption. However, we demonstrate at least one way to improve the jitter characteristics of an existing ATE system (see Sections 2 and 3).

Random jitter (RJ) is particularly troublesome because it is "unbounded". This means that there is a finite probability of error for ANY measurement. Luckily this error probability is exponentially small as we move the sampling time away from the "average" logic transitions (i.e. into the middle of the data eye). Since there is a relationship between bit error rate (BER) and the timing distance from the average transition point (measured in number of standard deviations, σ), we can bound the timing error (for a specified BER) by guardbanding (allowing sufficient timing margin to accommodate the expected amount of jitter).

A typical requirement for today's multi-gigahertz systems is to support a BER=10^{-12} (one in a trillion). Recently even tighter BER requirements are sometimes required. For BER=10^{-12} we must allow for a timing error of about 14σ (clearly much more stringent than the traditional 6σ limits). If for example σ=10ps, then 14σ=140ps. At 5Gbps, guardbanding for this timing error uses 70% of the available bit period (or "Unit Interval," UI). Allowing for a 70% UI timing guardband would greatly reduce yields, and in most cases render the product unprofitable. The whole problem becomes even worse when we try to test many parallel data channels which must all be aligned in time at the DUT I/O. In some cases we may have hundreds of these signals. Statistically, it is clear that we will need even tighter timing requirements under this scenario. Luckily not all multi-channel communications protocols require channel-to-channel timing alignment. PCI-express, for example, treats the timing of each channel almost independently (with each carrying its own encoded clock reference). Nevertheless, we must still allow for the 14σ timing margin when testing these channels.

Therefore, at data rates of a few gigahertz the value for σ needs to be in the picosecond range (smaller if possible). In practice, achieving σ~1ps in a complex multi-gigahertz ATE has been extremely challenging. Once introduced within the system it is very difficult (although not impossible) to reduce jitter. So usually jitter is tackled at its source by starting with a very clean (low-jitter) timing reference oscillator. The signal is then distributed as carefully as possible within the ATE, minimizing the additional jitter picked up by the signal before eventually reaching the DUT. Therefore great engineering effort and instrumentation costs are usually required to achieve this level of picosecond jitter. Any effective technique that can be used to reduce jitter is highly desired. We will present such a method in the Section 2, and show experimental demonstrations of this technique in Section 3.

Because testing at multi-gigahertz frequencies is so dependent on high-accuracy timing signals, it may be necessary to utilize more ATE resources for the most critical signals (such as clocks and timing references). For example, in our applications we have added multiplexing and demultiplexing modules to loadboards in order to synthesize multi-gigahertz test patterns using multiple 1Gbps ATE channels **[4-7]**.

An example testing application is shown in **Fig.1a**, using several "Driver" and "Receiver" modules [7] to provide over 80 multi-Gbps differential channels in a

978-3-9810801-2-4/DATE07 © 2007 EDAA

combination of HyperTransport (1.6Gbps per channel) and PCI-express (at 2.5Gbps per channel). The ATE jitter level (typically 4ps-8ps) is actually quite good for such a large complex system, and is suitable for testing at 1 Gbps. However, this jitter level is not sufficiently low for our testing needs in the 3 to 10 Gbps range. Therefore we use an external, low-jitter (~1.4ps) RF clock source (Agilent 81133A) to provide a programmable timing reference to each of the modules. This level of jitter is a factor of 3 to 6 smaller than that produced by the ATE and meets the needs for testing in the 1 to 5 Gbps range (testing at 10 Gbps my require even better performance).

In our example application, the external reference signal is used for various purposes by the different modules. In each case, this reference signal limits the timing accuracy that can be obtained by the module. While the external source does provide the needed low-jitter timing reference, it has some practical drawbacks for the production test environment where external instruments are at best awkward. It is also difficult (but not impossible) to obtain precise synchronization (and skew adjustment) between the ATE and the external signal generator across a wide range of frequencies. Therefore, we would prefer to obtain the timing reference signals *directly* from the ATE, if we could reduce jitter to acceptable levels (~1ps or less). The method described in this paper accomplishes this objective. It allows us to replace the external low-jitter source with directly-programmable ATE channels.

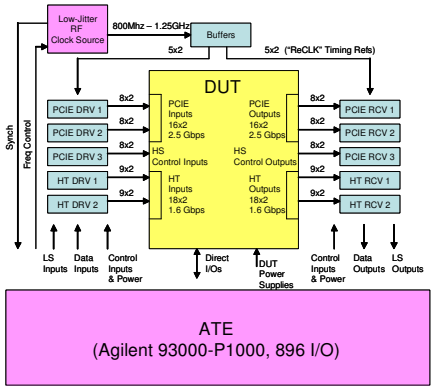

Fig.1a – An example multi-GHz testing application, requiring a low-jitter timing source.

A photograph of the bottom side of the example load-board is shown in **Fig.1b**. Five water-cooled receiver modules are seen mounted on the left side of the load-board, and five driver modules are on the right side. During production testing this entire assembly is flipped over and connected to the ATE test-head. The DUT test socket is on the top side of the load-board (not visible in the photo).

2 Jitter Reduction Method (Theory)

It is commonly thought that once jitter is introduced into a system, it cannot subsequently be eliminated or reduced. However, we have found that under certain conditions it is possible to make a trade-off between the level of jitter and the amount of ATE hardware resources used to produce the signals. Our method (described next) provides a way to get the needed jitter reduction at the cost of additional test hardware (it uses multiple ATE channels). Our strategy is to leverage the existing ATE resources to produce the low-jitter timing reference signals described in the example above. The method is generally applicable whenever low-jitter signals are required, and so it is important beyond the scope of this specific application that motivated its development.

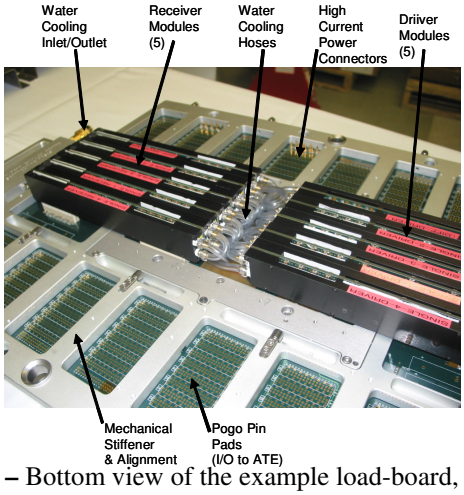

Fig.1b – Bottom view of the example load-board,

First consider a well-known effect that is observable on most digital sampling oscilloscopes (DSO). If we observe any repetitive signal at a fine enough timing scale we typically see either a "jittery" transition or (when selecting a longer persistence display mode) a wide transition band, as shown in **Fig2a**. The width of this jittery distribution band provides a crude estimate of total jitter (TJ). On the other hand, a common "trick" is to set the oscilloscope display to an "averaging" mode. Usually this will result in a much "nicer" display (narrower trace width on the screen) as shown in **Fig.2b**, and generally make it easier to get repeatable timing measurements. It is deceptive that the much of the jitter seems to "disappear." The signal is still jittery, it is just that the random components of this jitter are, in a sense "averaged out" by the sampling display process. What we are seeing on the screen is the result of many measurements taken over a long period of time. So we cannot use this effect directly to produce a useful real-time low-jitter signal. On the other hand it does suggest a possible approach.

If we analyze what is happening in the DSO, we see that the apparent jitter reduction occurs because the instrument is averaging widely-separated repetitions of the "same" signal. However, each repetition is not really the "same" signal, but rather another re-run of the test. Each time the scope is triggered (usually at a much slower rate than the data rate), the waveform is sampled at a completely different point in time (a different bit

period). By synchronizing the scope trigger to a multiple of the bit period, it appears as though we are sampling the "same" signal.

So, how can we utilize this effect to generate a truly real-time averaging effect? If instead of *serially* averaging a repetitive signal, as in the example above, we instead average two or more *parallel* (synchronized) signals, then we can obtain the desired real-time effect. An ATE with many channels can provide these parallel signals. We can use a simple resistive network to combine and average (in real time) these signals (see below).

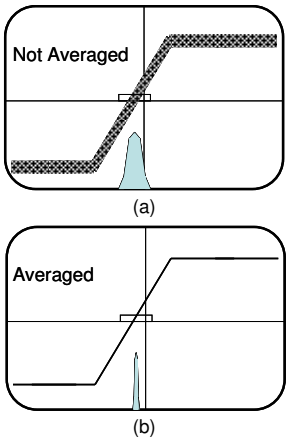

Fig.2 – DSO waveform averaging effect.

The basic (two input) configuration for this jitter-reduction method is shown in **Fig.3**. Here two synchronized signals (perhaps timing references or clocks from the ATE) are input to a simple resistive averaging network. The network is designed using standard impedance-matching techniques and resistive-divider configurations to produce an output which is proportional to the voltage average of the two input signals (see later discussions for details). A fast logic buffer is used to recover the full amplitude swing. For simplicity, two single-ended input signals are shown. However in practice we use differential-input buffers, and usually more than two inputs. In this figure, the output waveform C is depicted as having lower jitter than either input A or input B. This jitter reduction is expected for the same reason that the DSO averaging mode shows lower jitter. However, now the averaging occurs in real-time, and the output signal C actually has lower jitter than either of the two input signals.

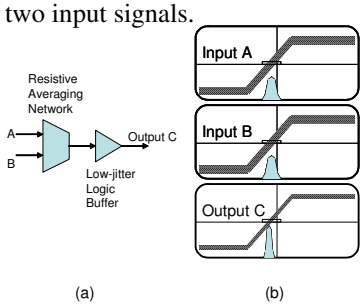

Fig.3 – Basic Jitter-Reduction Configuration (a), and I/O waveforms (b).

Given this basic jitter reduction effect, let's now quantify the expected jitter improvement. First let's be clear about the assumptions:

(1) We assume that the inputs are roughly synchronized (average transitions are synchronized), however
(2) Each input signal has *independent* random characteristics (i.e. NOT correlated).
(3) The resistive averaging network produces the instantaneous voltage-average of the two input signals.
(4) The inputs signals have finite rise-times that are comparable to or longer than the total jitter (TJ).
(5) The input signal edges are approximately linear near the 50% crossing point.
(6) The logic buffer has an effective input threshold near to the 50% crossing point of the inputs.
(7) For mathematical simplicity we will assume that RJ is similar for all the inputs (not strictly required).
(8) For simplicity we neglect the fixed finite delay of the buffer.
(9) For simplicity we assume that the jitter distribution of each signal is "normal" (Gaussian).

In assumption (1) we recognize the situation present in most ATE, where the entire system is driven with a common master clock (aside from multi-clock modes available on some ATE).

Assumption (2) is very critical to the mathematical analysis which follows, and is the primary reason why the method is successful at reducing jitter. Basically the method relies on the independent random jitter of one signal to "average-out" part of the jitter in another input. Actually if the ATE signals were perfectly synchronized, then there would be no opportunity to exploit this effect. If both inputs were *perfectly* synchronized (with correlated jitter), then the resulting output jitter distribution would be exactly the same as that of the individual inputs (no improvement). In practice we have found that the assumption of independent random jitter seems to be valid for the ATE channels we used in our experiments.

Note that the effects of assumptions 3,4,5,6 together results in the circuit acting not only as a voltage-averager but also as a time-averager of the input transitions. This effect simplifies the mathematical analysis that follows.

Given the assumptions described above, we can start the mathematical analysis by writing the probability density functions $P_A(t_A)$ and $P_B(t_B)$ for the two input signals based on the well-know Gaussian function, with their means set at t=0:

$$P_A(t_A) = (1/\sigma_A\sqrt{2\pi})\exp(-(t_A)^2/2\sigma_A^2)$$
$$P_B(t_B) = (1/\sigma_B\sqrt{2\pi})\exp(-(t_B)^2/2\sigma_B^2)$$

Where t_A and t_B are the 50% crossing times, and σ_A and σ_B are the standard deviations for the jitter distributions of inputs A and B respectively.

We can further simplify the mathematics by assuming that the two standard deviations are equal ($\sigma_A = \sigma_B = \sigma$). Then we have:

$$P_A(t_A) = (1/\sigma\sqrt{2\pi})\exp(-(t_A)^2/2\sigma^2)$$
$$P_B(t_B) = (1/\sigma\sqrt{2\pi})\exp(-(t_B)^2/2\sigma^2)$$

Using the resistive averaging network, the output timing (t_C) is the average of the two input timing values t_A and t_B. So t_C is given by:

$$t_C = (t_A + t_B)/2$$

The variance, σ_C^2 for the output waveform is defined as:

$$\sigma_C^2 = \int_{-\infty}^{+\infty} (C_0 - t_C)^2 P_C \, dt_C$$

Since we have assumed that $A_0=B_0=0$, we can conclude also that $C_0 = 0$.
Therefore we have:

$$\sigma_C^2 = \int_{-\infty}^{+\infty} (t_C)^2 P_C \, dt_C$$

We can substitute $P_C=P_AP_B$ and $t_C = (t_A + t_B)/2$ and integrate over the two independent variables (t_A, t_B). Then the output signal variance becomes:

$$\sigma_C^2 = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} [(t_A + t_B)/2]^2 \, P_AP_B \, dt_A dt_B$$
$$= (1/4) \int_{-\infty}^{+\infty} (t_A^{\,2} + 2t_At_B + t_B^{\,2}) \, P_AP_B \, dt_A dt_B$$
$$= (1/4) \left[\int_{-\infty}^{+\infty} (t_A^{\,2}) \, P_A dt \int_{-\infty}^{+\infty} P_B dt + \int_{-\infty}^{+\infty} (t_B^{\,2}) \, P_B dt \int_{-\infty}^{+\infty} P_A dt \right] + (\text{zero terms})$$

$$= (1/4) \left[\int_{-\infty}^{+\infty} (t_A^{\,2}) \, P_A dt + \int_{-\infty}^{+\infty} (t_B^{\,2}) \, P_B dt \right]$$

because $\int_{-\infty}^{+\infty} P_A dt = 1$

$$\sigma_C^2 = (1/4) [\sigma_A^2 + \sigma_B^2]$$

Therefore,

$$\sigma_C = (1/2) (\sigma_A^2 + \sigma_B^2)^{1/2} = 2^{-1/2} \, \sigma = 0.707 \, \sigma$$

(again assuming that $\sigma_A = \sigma_B = \sigma$).

This shows that the output jitter is reduced to about 71% of the input jitter values (for two inputs)!

It is possible to generalize this result for the case of multiple (N) inputs as follows:

$$\sigma_{Output} = (1/N) [(\sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \ldots + \sigma_N^2)]^{1/2}$$
$$= N^{-1/2} \, \sigma$$

(if all inputs have the same standard deviation).

Table I indicates some representative values for the theoretical jitter reduction factor, depending upon the number of inputs, and assuming that all have the same amount of random jitter.

Table I – Theoretical jitter reduction values as a function of the number of inputs (N).

N	σ_{Output}
1	100% σ
2	71% σ
4	50% σ
8	35% σ
16	25% σ

Therefore, a reduction of jitter by a factor of 2, 3, or perhaps 4 or more seems feasible. From a practical perspective, there will be a point of diminishing returns since each additional input uses expensive ATE resources (channel pin electronics). There is also a practical limitation due the fact that the circuit used to combine the inputs itself adds some amount of jitter. As the number of inputs increases, it is likely that this added jitter will also increase, and limit the effectiveness of the jitter-reduction circuit. Also, the assumption that all inputs are nominally synchronized means that considerable calibration effort is required to measure and adjust all the input delay values. Higher values of N will therefore use more tester hardware and require longer calibration times.

As shown in the next section, we have tried N=2, N=4, and N=8 using two experimental circuits, and have found surprisingly close agreement with the predicted jitter reduction values. It is tempting to extrapolate to higher values of N. However we suspect that the practical limitations described above may prevent significant improvements beyond N=8 or so.

3 Experimental Validation of Jitter Reduction

To validate the predicted jitter reduction values described in the last section, we constructed an experimental circuit with 4 inputs that could be connected either to external instruments or to four channels of an ATE (an Agilent 93000, P-1000). A very low-jitter (σ ~100femtoseconds) SiGe Bipolar buffer was used to recover the full logic swing (in this case about 400mV) following the resistive averaging network. The circuit design shown in **Fig.4** was named "Clock Cleaner 1" and was fabricated using standard multilayer printed circuit board technology with controlled-impedance traces. Surface-mounted microwave chip resistors were used for the averaging network. The series resistor value (Rs) of 20 Ohms was chosen in order to provide a 50-Ohm impedance match into each of the 4 input ports (A,B,C,D). This allows the circuit to be driven from either the 50-Ohm source impedance ATE pin electronics or by standard test instruments. The resistive averaging circuit produces voltages proportional to the pair-wise averages of (A,B) and (C,D). These are used as the differential inputs for the SiGe buffer.

To accomplish the desired effect, the signals input to A and B should be nearly identical, and approximately synchronized (except for their independent jitter characteristics). Signals input to C and D should be the complements of A and B, again with independent jitter.

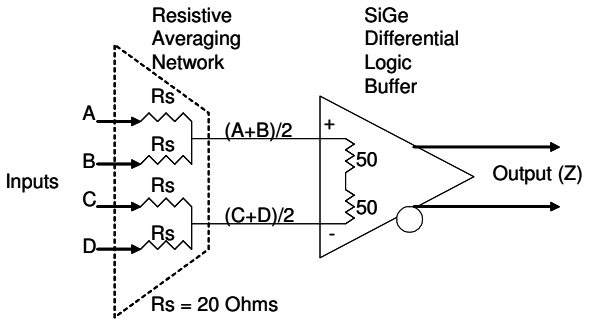

Fig.4 – "Clock Cleaner 1" (4 input prototype) circuit.

The first test configuration is shown in **Fig.5**. Here two Agilent 81133A signal generators are phase-locked using a 10 MHz reference signal. Each instrument then uses its own internal oscillators and timing generators to produce the programmed differential signals, which are then connected to the 4 inputs of Clock Cleaner 1. An Agilent 33250A AWG is used to provide a ~50MHz white noise voltage source to the signal generators for injecting jitter. The 81133A instruments use the random input voltage source to modulate their output signal delays. By adjusting the amplitude of the voltage noise we were able to control the amount of injected timing jitter. We also adjusted cable lengths and programmed delay settings in order to obtain non-correlated random jitter between the two signal generator outputs. Because the differential pair from the generator comes from a common source, its two halves are expected to have highly-correlated jitter characteristics. Therefore, even though we have 4 physical input ports, there are really only 2 independent jitter distributions in this first measurement configuration (N=2 for this setup). For the initial jitter measurements we used a Tektronix TDS6154C (15 GHz real time oscilloscope with 8 GHz differential input amplifier). This instrument has time interval error (TIE) and jitter histogram analysis capabilities.

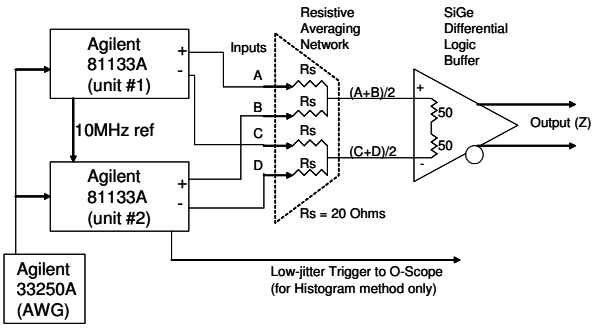

Fig.5 – Test configuration for Clock Cleaner 1.

Tables II and **III** provide the initial experimental results, showing the measured input and output jitter values as well as the theoretical predicted values. Because the TIE method does not require a separate trigger input to the TDS6154C, the measured values in Table II are generally more accurate than those in Table III (based on the traditional histogram method). The histogram method (Table III) consistently produces larger than expected measurements, perhaps due in-part to its reliance on an external trigger. Considering the difficulties in making these small-value measurements, we believe that they demonstrate good agreement with those predicted by theory.

Table II – Measurement results for Clock Cleaner 1, using the *TIE* method.

σ_{AC}	σ_{BD}	$\sigma_{Z(Measured)}$	$\sigma_{Z(Theory)}$	$\sigma_{Z(Measured-Theory)}$
3.8ps	1.5ps	2.3ps	2.04ps	0.26ps
7.6ps	7.6ps	5.2ps	5.37ps	-0.17ps
14.0ps	14.0ps	9.3ps	9.89ps	-0.59ps
21.0ps	21.0ps	15.3ps	14.85ps	+0.45ps

Table III - Measurement results for Clock Cleaner 1, using the *histogram* method.

σ_{AC}	σ_{BD}	$\sigma_{Z(Measured)}$	$\sigma_{Z(Theory)}$	$\sigma_{Z(Measured-Theory)}$
9.8ps	9.8ps	7.7ps	6.93ps	+0.77ps
12.5ps	12.5ps	9.8ps	8.84ps	+0.96ps
18.0ps	18.0ps	14.0ps	12.7ps	+1.30ps
23.0ps	23.0ps	19.0ps	16.3ps	+2.70ps

For the final demonstration of our original Clock Cleaner 1, we replaced the two Agilent signal sources with four ATE channels and measured both the input and output random components of jitter (RJ). The jitter values are consistently about 4.1ps across all four inputs. In this test configuration we did not have a convenient way to adjust the amount of random jitter for the input signals (this was fixed by the Agilent 93000 ATE). Because we now have 4 independent signals (N=4), the expected jitter for the clock cleaner output is 50% of the input **σ** values. Remarkably ***the measured output jitter value is 1.79ps, which is within 0.26ps of the predicted value of 2.05ps.*** This clearly shows that we can substantially reduce jitter for even low-jitter ATE signals using this new method.

In a more recent experiment we built a new "Clock Cleaner 2," with 8 inputs. A schematic for Clock Cleaner 2 is shown in **Fig.6**. Some other features were added to further minimize the jitter characteristics. These features included improved (higher-bandwidth) coaxial connectors, use of low-loss high-frequency dielectric materials in the PCB construction, double-buffering of the output signals, and a 1:4 fanout output buffer (providing four synchronous differential output ports). Clock Cleaner 2 is intended to be integrated into an active load-board (as in Fig.1) where multiple modules can utilize the "cleaned" clocks as timing references.

Fig.7 shows a comparison of the measured input and output jitter distributions for the new Clock Cleaner 2. The top figure is typical of one of the eight input signals, and shows about 4.1ps for **σ**. With 8 similar inputs, we expect a jitter reduction to about 0.35**σ.** Indeed the bottom of Fig.7 shows that the Clock Cleaner 2 output has about this much jitter (~1.3ps). Notice that the two

plots have different horizontal scales, so the reduction in jitter is not obvious at first glance.

The TDSJIT3 statistical analysis tools provide an even more detailed and accurate measurement values. The Clock Cleaner 2 output peak-to-peak *total* jitter at BER=10^{-12} was only about 20ps. This was about 1/3 that of the inputs total jitter values (typically ~59ps). Test results found random jitter standard deviations of 3.6 to 3.9ps for the 8 independent inputs, with an average input jitter value is about 3.74ps. The measured output jitter is 1.27ps, which is ***within 40 femtoseconds*** of the predicted value ($\sigma_{Z(Theory)}$=0.35x3.74=1.31ps). We found this to be exceptionally good agreement with theory. The method gives a practical way to reduce random jitter (approaching 1ps) in a source that is directly controlled by the ATE (using no external instruments).

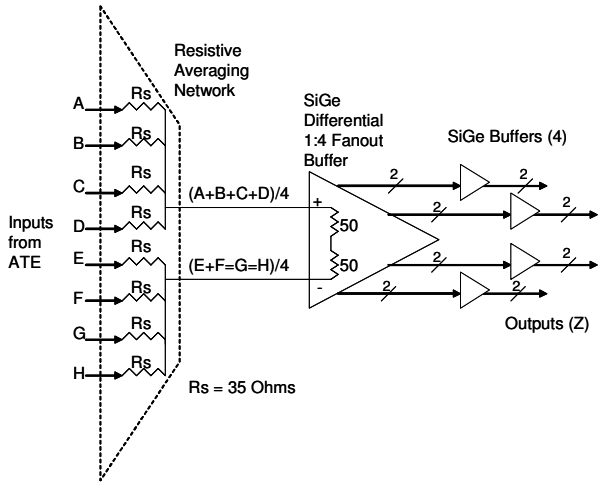

Fig.6 –ClockCleaner2(8-input prototype)circuit diagram.

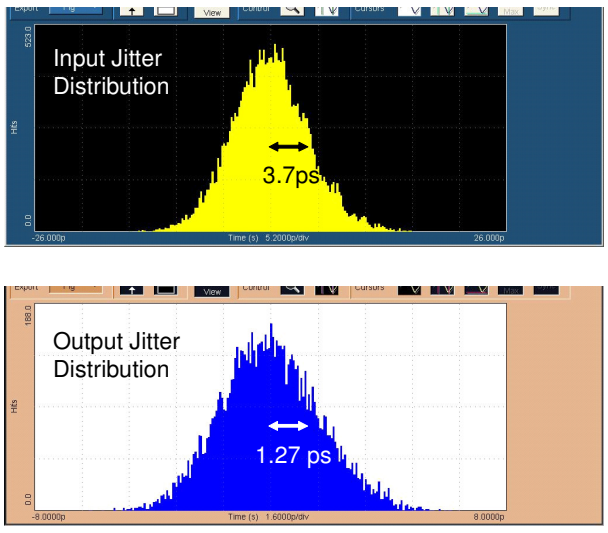

Fig.7 – Clock Cleaner 2 jitter I/O histograms - Typical input jitter distribution (top), and output distribution (bottom). Note the difference in horizontal scales.

4 Conclusions

We have presented a method for synthesizing a low-jitter signal by real-time averaging of multiple signals each with higher levels of jitter. This technique has wide applicability to many systems where low-jitter is required, including multi-gigahertz ATE (with picosecond jitter requirements). Several measurements were made on two experimental circuits which clearly demonstrated the feasibility of this method and showed very close agreement with theory. For an example ATE application, we demonstrated reduction of signal jitter by about 65% (from 3.74ps to 1.27ps) using an 8-input version of this jitter reduction method. In this case the jitter reduction agreed with theory to within 40 femtoseconds. In the ATE application the low-jitter signal has the additional advantage of being under the direct control of the ATE, so that both frequency and delay (phase) can easily be adjusted as needed for a particular test. The low-jitter signals are also inherently synchronized with the rest of the ATE without the need for phase-locking or external instruments.

5 Acknowledgements

This work was conducted as a joint R&D project between Georgia Tech and IBM, Canada**.** The system-level experiments were conducted using equipment at the IBM, Canada facility in Bromont.

6 References

[1] M. Shimanouchi, *"Periodic Jitter Injection with Direct Time Synthesis by SPP ATE for SerDes Jitter Tolerance Test in Production,"* Proc. of the Intl. Test Conf. (ITC'03), pp. 48-57, 2003.

[2] T. Yamaguchi, M. Soma, M. Ishida, M. Kurasawa, H. Musha, *"Effects of Deterministic Jitter in a Cable on Jitter Tolerance Measurements,"* Proc. of the Intl. Test Conf. (ITC'03), pp. 58-66, 2003.

[3] K. Taylor, H. Lin, A. Chong, E. Chan, M. Soma, H. Haggag, J. Huard, J. Braatz, *"CMOS Built-in Test Architecture for High-Speed Jitter Measurement,"* Proc. of the Intl. Test Conf. (ITC'03), pp. 67-76, 2003.

[4] D.C. Keezer, D. Minier, M.C. Caron, *"A Production-Oriented Multiplexing System for Testing above 2.5 Gbps,"* Proc. of the IEEE Intl. Test Conf. (ITC'03), pp.191-200, Charlotte, Oct. 2003.

[5] D.C. Keezer, D. Minier, M. Paradis, F. Binette, *"Modular Extension of ATE to 5 Gbps,"* Proc. of the IEEE Intl. Test Conf. (ITC'04), pp.748-757, Charlotte, Oct. 2004.

[6] D.C. Keezer, D. Minier, M.C. Caron, "Multiplexing ATE Channels for Production Testing at 2.5 Gbps" *IEEE Design and Test of Computers*, Vol.21 No.4, pp. 288-301, July/August 2004.

[7] D.C. Keezer, D. Minier, P. Ducharme, "Source-Synchronous Testing of Multilane PCI Express and HyperTransport Buses," *IEEE Design and Test of Computers*, vol. 23, no. 1, pp. 46-57, January 2006.