Spectral Error Density Analysis – A New Approach To Identify Jitter Sources in SOC-Devices

Bernd Laquai
Agilent Technologies
Herrenbergerstrasse 130, Boeblingen, Germany
E-mail: bernd_laquai@agilent.com

July 17, 2003

Abstract
Massive parallel embedding of high-speed IO ports running at multiple Gigabits per second is certainly the most prevalent trend these days to match the lagging IO bandwidth of SOC-devices to the constantly growing performance of the digital core logic. However, crosstalk from millions of simultaneously switching digital transistors, injects jitter into the sensitive analog high-speed IO ports and substantially eats up the timing margins of the sub-nanosecond bit intervals available for transmission. Therefore, managing this jitter-injecting crosstalk is the toughest challenge to be successful with the new IO technology.

IC-testers (ATE) as well as parallel bit error rate testers (BERTs) are readily available able to drive high-speed data and to compare it to expected data at-speed. This equipment is required to verify the critical parameters such as the bit error rate (BER) and the amount of jitter to confirm compliance with the key standards or proprietary specifications. But which strategy should you use when your tests fail and indicate too much jitter and excessive bit error rates? How can you identify the major jitter sources?

Evaluating the jitter histogram or the bathtub BER characteristic is certainly the first step but honestly, does it tell you much about the individual sources? A much more sophisticated approach is possible with your existing capture-and-compare equipment that finally leads you to a spectral decomposition of the jitter. If you know the frequencies of the individual jitter components you can trace the jitter back to the frequencies of the clocks you are using in your digital core and localize the crosstalk problem. All what you need to do is to analyze the error density data in the frequency domain using your existing at-speed capture-and-compare equipment.

1. The need for test system integrated jitter debugging

Imagine you have embedded a PCI-Express interface into your highly complex digital SOC device. The interface is running at 2.5Gbit/s data rate on 16 lanes and in contrast to the standalone testchip, you may have used for initial characterization of just the macrocell, the embedded IO cells show excessive jitter. What can you do to identify the jitter source and to fix the problem as fast as possible and with a minimum of further silicon cuts? You suspect a power supply crosstalk from your digital SOC core-clock but hooking up a spectrum analyzer on the high-speed IO outputs of your ATE loadboard carrying your 1000 pin SOC device just to identify the jitter frequency looks pretty difficult and expensive. Relax! - You don't need to create an additional sensitive connection to additional and costly equipment. You can re-use your existing equipment to even do spectral jitter analysis. There is a simple and cheap trick you can use to exploit the power of your high-speed IO tester channels. Here's how.

Many companies, having done purely digital SOCs in the past, are currently forced to execute a dramatic change in the IO technology that makes the above scenario very probable. The new IO interface technology runs at many Gigabits per second and uses sensitive analog circuitry, squeezing even advanced digital CMOS processes to the limit to achieve a maximum of bandwidth. The goal is just to match the huge processing power of today's digital cores. Phase locked loops (PLLs), are used for on-chip gigahertz transmit clock synthesis and receiver clock and data recovery. Waveform shapers, amplifiers and equalizers are the key analog building blocks that make the high-speed transmission on lossy low-cost FR4 circuit boards possible. These analog circuits were developed by the communications-focused chipmakers years ago using small standalone physical layer transceiver devices for standards such as SONET or IEEE 802.3 Ethernet. It was possible to characterize these single port high-speed communication devices on single channel bench equipment without the need to keep a digital core running simultaneously. Single channel bit error rate testers (BERTs), high bandwidth oscilloscopes and spectrum analyzers were used to verify and debug the parametric performance required to comply with the standards. Jitter, bit error rate and eye opening were the most critical parameters.

With the advent of nanometer CMOS technologies and the leverage of high-speed IO technology to computational applications such as microprocessors and PC/Server chipsets, this situation has

dramatically changed. Your goal today is to embed hundreds of high-speed IO ports into large digital host designs also in order to profit from the bandwidth advantage of this new IO technique and to better balance your core and IO performance. You run huge digital cores at several hundred megahertz or even gigahertz using minimum skew clock distribution in multiple cores and several clock domains. You embed the high-speed IO interfaces massively in parallel and place them in the immediate vicinity of the digital cores on the same substrate. This creates a completely different noise floor in terms of supply crosstalk and micro-field impact on the sensitive analog IO circuitry than compared to the standalone physical layer transceiver device of the past. Furthermore you use plain CMOS processes tweaked for highly integrated digital logic to build complex heterogeneous SOCs containing the high-speed IO interfaces pretty different from what was formerly done in an almost ideal form on the small specialized GaAs or SiGe die area used for the standalone single port device. Process variations that have only a minor influence on digital strongly affect the analog performance and the signal integrity parameters of your high-speed IO cells. The results are less robust initial designs and low yield, mostly caused by parametric failures at the high-speed IOs. Every additional silicon cut affects the time to market. Fast parametric debugging and rapid yield learning are therefore crucial for your success and finally determine if you will become profitable with your product.

Controlling jitter is the number one factor for success when embedding high-speed IO cells. In an environment where multiple serial point-to-point connections (so-called lanes) run asynchronously in parallel, jitter replaces the key setup-and-hold time specification known from traditional parallel interfaces. Jitter can have many different sources in your design, is analog in nature and you can hardly simulate it for a complete huge SOC device running in mission mode taking all possible crosstalk paths into account. The traditional way to classify jitter is with statistical means. The Gaussian probability density distribution (PDF) is a widespread model for random jitter as it occurs at semiconductor junctions and is quantified by its rms-value. Deterministic and periodic jitter caused by clock crosstalk appears as a bi-modally distributed density in case of a single binary clock source and is quantified by its peak-to-peak value. But the probabilistic view does not help you much, except for quantification of the jitter. A histogram on a sample set of signal transitions approximates the jitter probability density function and gives you at most some insight into how many different sources you have or what randomness is behind the jitter but it does not help you much in identifying the source.

What significantly helped the communication transceiver designer with debugging for deterministic jitter sources in the past was the analysis of the phase noise in the spectral domain. Since jitter is a variation of transition events versus a regular timing grid you can trace it back to phase modulation and the concept of phase noise measurements using a clock-like data signal. According to modulation theory the sinusoidal phase modulation of a sinusoidal carrier signal with small modulation depth results in a spectrum showing two side peaks to the left and right of the carrier. You can extract the jitter frequency from the distance of the side peaks to the carrier. Looking at band limited random phase fluctuation as a source of jitter modulation you'll see a noise-skirt around the carrier (the phase noise characteristic). It is actually a left and right sideband to the carrier representing an infinite overlay of modulation sinusoids. In the past the communication device designer chose a spectrum analyzer to perform this spectral analysis task. He performed a spectrum analysis at the high-speed transmit output of his device while providing a simple pattern to the parallel interface of his standalone transceiver to obtain a clock-like sequence at the high speed transmit output as a substitute for a sinusoidal carrier. The jitter frequency information was extremely helpful therefore many communication device designers spent the high effort to do this additional costly connection to the spectrum analyzer in order to get one step further in debugging their device for deterministic jitter sources.

Looking at the situation today, however, along with the task to debug a highly complex SOC device for parasitic jitter sources you typically have the requirement to exercise the complete device in mission mode on a characterization test system with limited external access to the many high-speed IO pins on the ATE loadboard or the evaluation system board. In the world of SOC testing the test environment has to be neat and clean with a minimum of additional parts that could affect the repeatability and signal integrity of a test involving the high-speed IOs. Leading edge, high end tester channels are carefully connected through impedance matched transmission lines and are used to capture and compare on a functional basis resulting in bit error rate characteristics versus strobe and threshold position (so-called bathtub curves or eye diagrams). You typically extract jitter parameters from the horizontal opening of a bathtub curve or an eye diagram. You can also calculate the jitter histogram from the error count while sweeping the strobe point of your channel comparator or from forming the derivative of the bathtub characteristic. However, as indicated before, jitter quantities or the histogram does not give you what you need for debugging: an indication of what the root cause of the jitter is. It's

Agilent Technologies

better to do a spectral analysis of the jitter to get an idea of the modulation frequency of the jitter source. This can probably be done using only your tester channel and not touching the transmission path of the connection. The spectrally decomposed jitter information can be compared to known clock frequencies in the core and would lead you directly to the jitter source. Well, even though it sounds like a miracle, converting your capture-and-compare tester channel to some sort of spectrum analyzer for spectral jitter analysis is really possible.

2. Spectral error density analysis using at-speed capture-and-compare equipment

When the high-speed IO technology was introduced the eye diagram measurement technique leveraged from the communication world may have surprised many engineers in terms of the plurality of parametric performance parameters you can extract from it. It is certainly one of the most important measurement techniques with respect to high-speed signal integrity. However, the construction of the data eye by nature removes any relationship to the time domain for anything that is larger than the bit interval and thus makes a spectral decomposition of jitter impossible.

You have to go one step back from constructing the eye diagram and think about the waveform seen by the high-speed comparator in your existing capture-and-compare equipment to discover another option for looking at the jitter in a way that preserves the spectral information. The error density you'll obtain when you compare during the threshold crossing points of the signal transitions contains the complete spectral information of the jitter.

Just to start a train of thoughts for explanation, imagine you would put the compare strobe not in the center of the data eye as usual but half a unit interval (UI) to the left from the sweet spot of the lowest BER into the crossover point of transitions. Assuming that there is always a small amount of random jitter contained in the data signal you will get an equally distributed error signal from your tester channel as the result of comparison to expected data. This is because in some bit intervals you will still capture the correct bit because the current random offset of the transition occurs also to the left as you offset your strobe point. But in other bit intervals you will sample the adjacent bit that may be different from the currently expected bit because for this instance the shift of the current transition is to the right and exactly opposite to how you offset your strobe position. Due to the random behavior of the jitter and assuming random data content, both cases occur randomly mixed and you'll get a random error sequence with constant error rate or density from the comparison.

Now let's assume we have the situation of a strong deterministic jitter injection with bi-modal distribution caused from a cross-talking square wave clock with a given period (see figure 2.1). According to the jitter amplitude, transitions will be offset and the bit interval will shift during one half of the jitter clock period to the left (indicated in blue) in the same direction that you intentionally offset your strobe. In the other half of the jitter clock period the received bit intervals shift to the opposite and wrong side (indicated red) regarding the expected data available for the actual right hand bit interval. Therefore the strobe sitting in the crossover of transitions falls inside the correct bit interval (right side) for one half of the jitter clock period (blue) yielding no error and it will fall outside the correct bit interval for the other half of the jitter clock period yielding an error when the adjacent previous bit was different. As a result we will see an error density that is non-zero for the positive half period of the jitter period and an error density of 0 for the negative half of the jitter period. Thus the periodicity of the jitter maps into a periodicity of the error density. A sinusoidal jitter injection would cause a similar result because the sine wave would get mapped to a square wave shaped error density when it changes sign. A spectral analysis of the error density signal already would give you the fundamental frequency of the jitter you are looking for, even though it may come along with some harmonics.

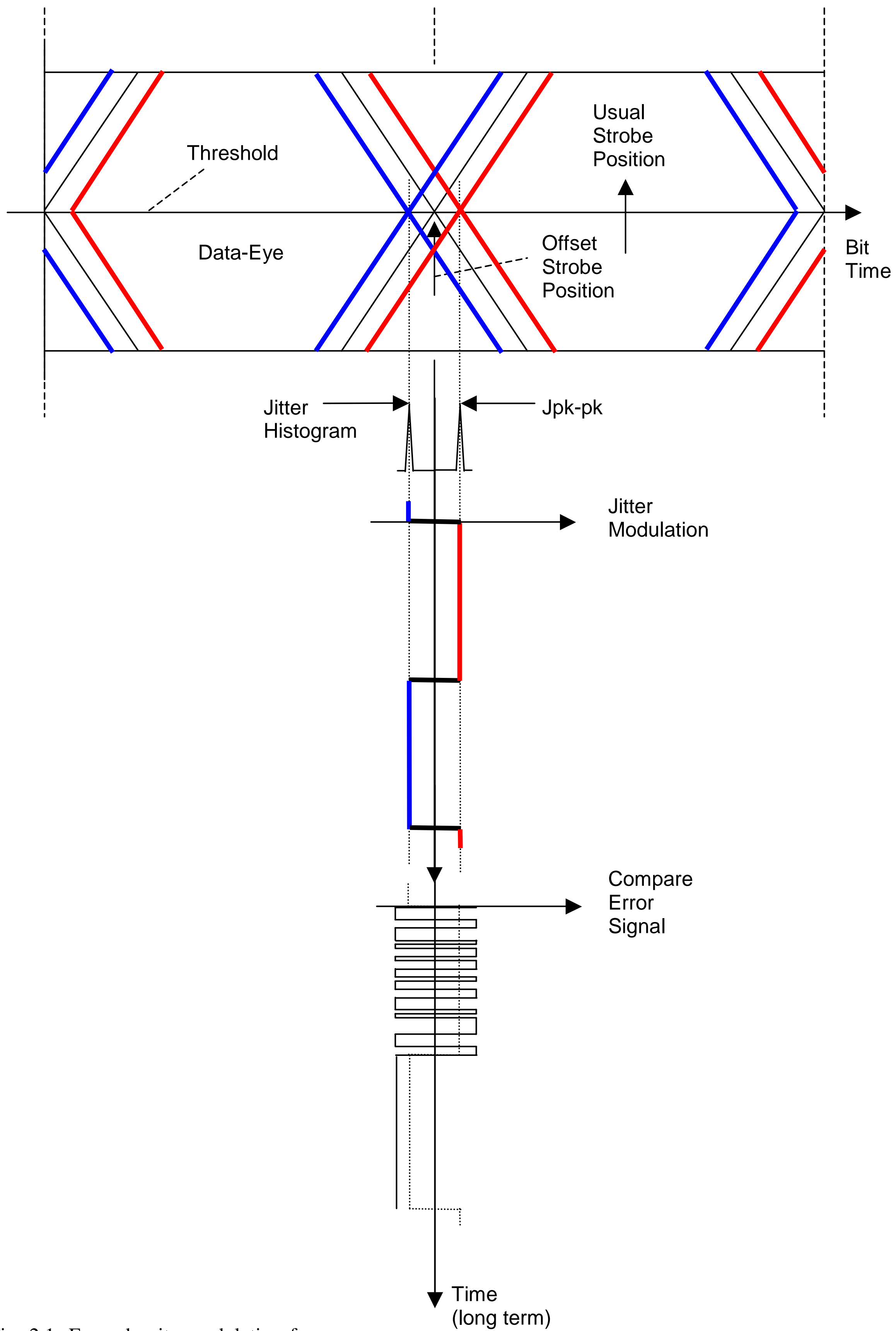

Fig. 2.1: Error density modulation for pure square wave jitter

Now let's change the situation in such a way that the amount of injected periodic jitter is not larger than the random jitter which typically should be the case for a real device you finally want to sell in the market. We also assume that the periodic jitter is sinusoidal in its modulation shape. In such a case the error density would also get modulated sinusoidally with the periodicity of the jitter source. During the one half period of the jitter sinusoid the bit intervals mostly shift to the left in the direction of your strobe offset but since the sinusoidal amplitude is small compared to the random jitter excursions it would just reduce the error density compared to the situation with random jitter only. During the other half period of the jitter sinusoid the bit intervals mostly shift opposite to your strobe offset and therefore the error density increases proportionally (see figure 2.2). Finally you will recognize that the error density now reflects the jitter modulation function in a linear fashion conveying the periodicity of the jitter without harmonics. This means that the relevant spectral information is mapped in a linear way into the error density signal. Therefore a spectral analysis of the error density signal yields all the spectral information of the jitter modulation domain and thus allows you to display the jitter energy versus the frequency. Any undesired clock crosstalk will pop up in the jitter spectrum as a discrete and distinct peak and any excessive random jitter that is band-limited appears as an elevated noise floor in the spectrum.

If the peak value of your periodic jitter component is larger than the rms-value of the default random jitter, harmonic distortion may occur but you still have the option to additionally inject artificial random jitter through the reference clock input of your PLL that synthesizes the high speed bit clocks of your IO ports. This helps to meet the requirement for the linear spectral mapping into the error density signal. Since it means to provide the modulated reference clock only once for all high speed IOs, this is not much of an effort. In contrast to the spectrum analyzer based phase noise measurement you are not restricted to a clock-like data signal, you can use real life random data. Therefore, the impact of pattern related periodic data dependent jitter (inter-symbol interference, ISI) also becomes visible in a spectrally decomposed way.

Random jitter energy spreads all over the spectrum, resulting in a fairly low spectral density per Hertz. Any periodic jitter hidden in the random jitter concentrates all its energy into discrete peaks clearly appearing out of the random jitter floor. Therefore the detection sensitivity of the method for periodic jitter components is extremely high and periodic jitter can be detected even when it is deeply buried in the random jitter and not visible in the Gaussian shaped jitter histogram.

By using this method, you can do the full spectral analysis on each capture-and-compare channel attached to the many high-speed IO ports in your device without changing sensitive transmission line connections. Simultaneously you may have your huge digital core exercised at full steam. This will show the worst case jitter impact. Therefore, this method gives you a full spectral jitter analysis capability per port. Isn't this is a big step towards localizing jitter sources in your SOC design?

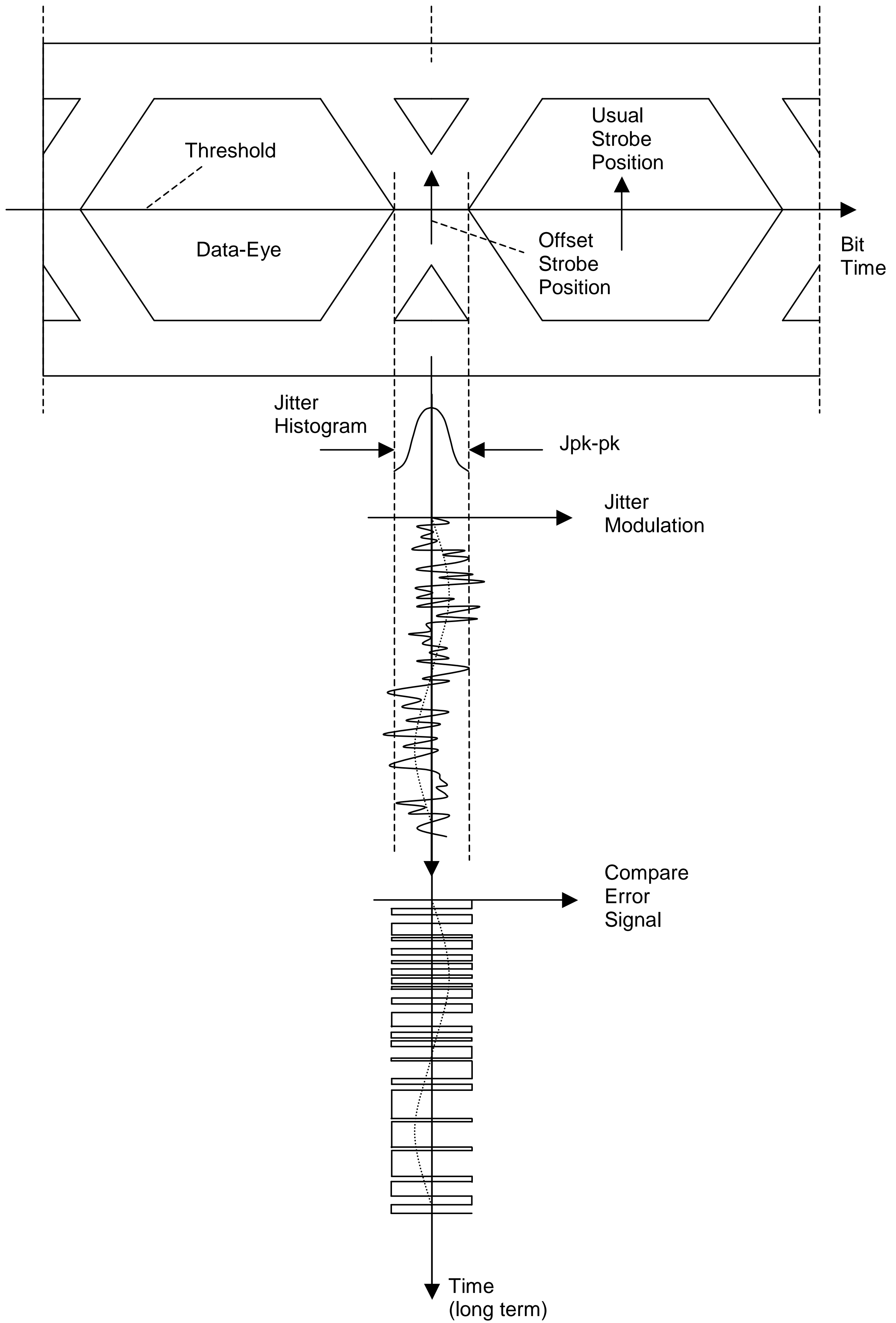

Fig. 2.2: Error density modulation for a mixture of random and sinusoidal jitter

3. A practical example

To visualize the power of this new analysis technique we used the reference clock to a 3.125Gbps high speed IO XAUI port to artificially inject periodic jitter to simulate an undesired type of jitter crosstalk.

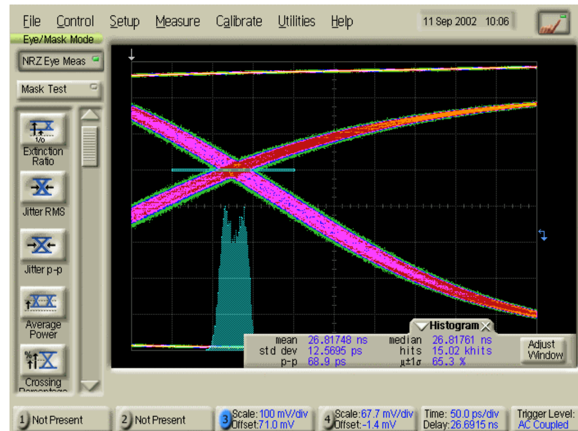

Fig 3.1: 1MHz sinusoidal jitter injected into the reference clock

Fig. 3.1 shows the 156.25MHz reference clock with a small amount of sinusoidal jitter of 1MHz frequency injected. The jitter histogram of the sinusoidal modulation is clearly visible. Fig. 3.2 shows the 3.125GHz transmit output signal using random data. Due to the additional intrinsic device jitter the small sinusoidal jitter is no longer visible, it is buried in the random jitter showing up with a Gaussian shaped distribution.

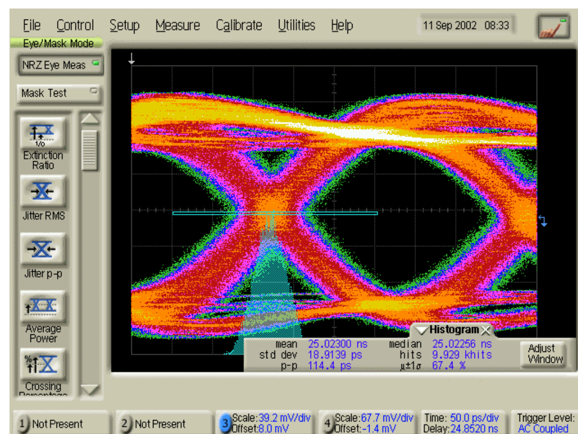

Fig. 3.2: Transmit output (random data at 3.125Gbps) containing the 1MHz sinusoidal jitter

Fig. 3.3 shows the result of comparing the jittered output data with expected data using the Agilent 81250 ParBERT instrument. The errors are indicated in red. What you can see here is the periodicity of the periodic jitter that leads to a periodic error density modulation. The fact that we use a 1MHz jitter modulation frequency and a device output rate of 3.125Gbps causes the error density maxima to reappear within 3125 bit intervals. Fig 3.4 finally shows the jitter spectrum resulting from analyzing the error density signal for example using the spectral jitter analysis software of the Agilent 81250 ParBERT that is an integral part of the 93000 XP3G ATE solution. The spectrum shows a strong peak at 1MHz and a much, much smaller random jitter noise floor that can also be analyzed using a logarithmic display. This highlights the enormous selectivity of the method for hidden periodic jitter.

To show the linearity of the mapping we used a square wave modulation waveform with an asymmetric duty cycle. Fig. 3.5 shows the phase modulation signal injected into the reference clock of the device. It has a 1us pulse width and a 10us period. Fig. 3.6 shows the data eye of the modulated reference clock. Due to the phase offset caused by modulation with the narrow pulses the left side of the data appears blurred. Looking at the high speed data output running random 3.125Gbps data you see the default random jitter with an elevated tail on the left side that corresponds to the blurred left side eye boundary caused by the narrow phase pulse. It is not possible to diagnose the jitter frequency and exact shape of the modulation function from the histogram. However, with the method described above you will be able to determine it. Fig 3.7 shows the jitter spectrum resulting from analyzing the data signal using the spectral jitter analysis software. Clearly you can see the spectral representation of the narrow pulse phase modulation fully according to theory, in the spacing of spectral lines as well as in the shape of the spectrum. This is a proof that the method is able to map multitone signals linearly in the spectral domain.

What you can also see is a parasitic peak as the result of a clock crosstalk problem. From looking at the jitter frequency of the parasitic peak finding the source of the originating crosstalk was fairly easy. In our example case it was unintentionally caused by a respective clock fraction derived by frequency division in a faulty clock generator. Therefore we were able to even debug our own measurement demonstration setup.

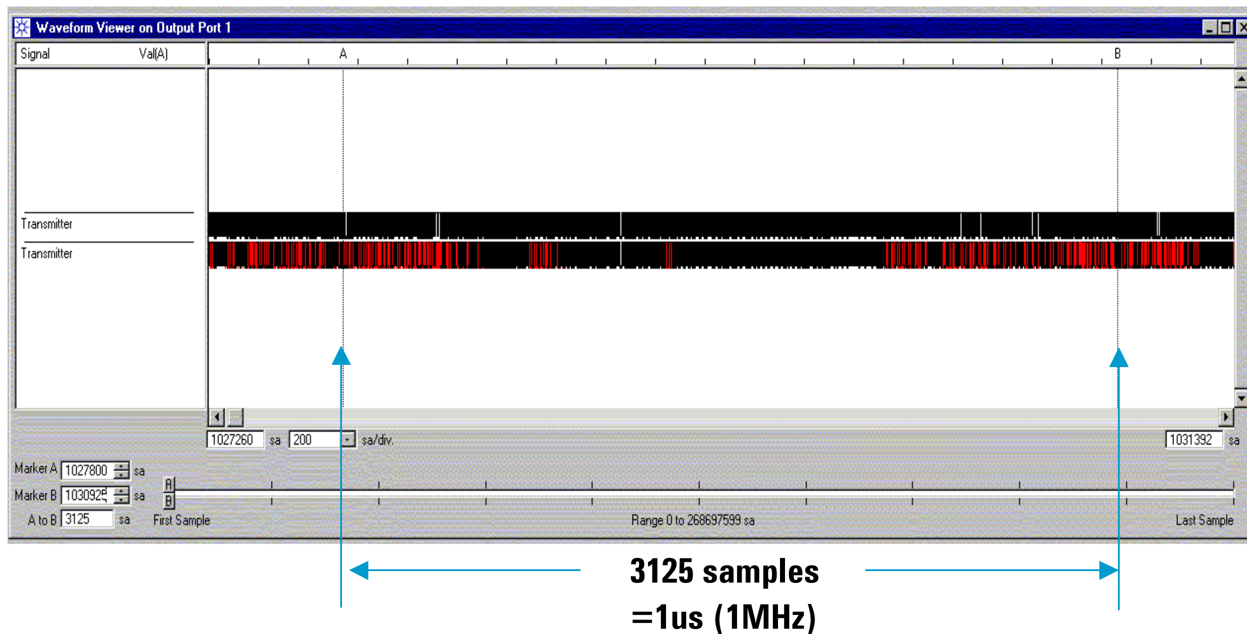

Fig. 3.3: Error density modulation as a result of the 1MHz sinusoidal jitter

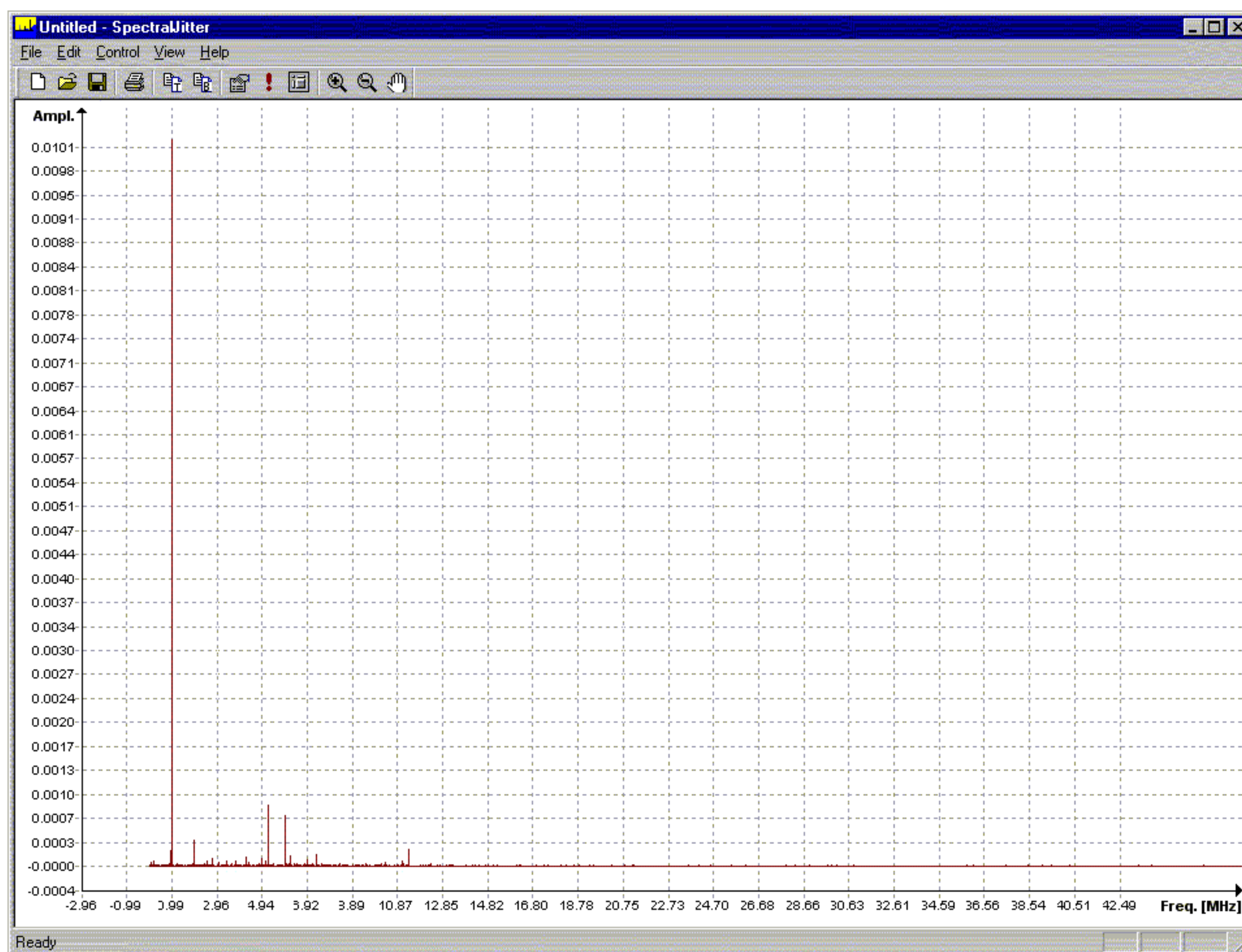

Fig. 3.4: Spectral analysis of the error density signal reflecting the 1MHz sinusoidal jitter

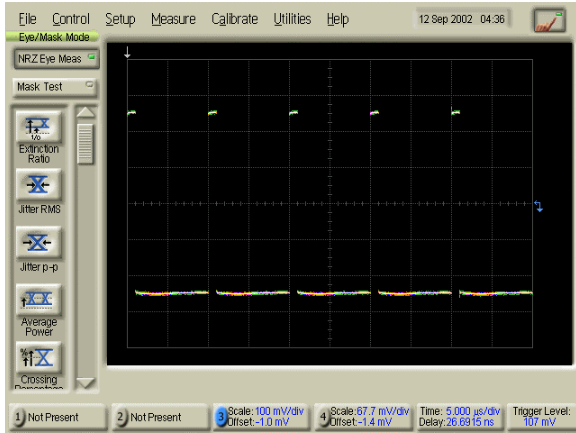

Fig. 3.5: Narrow pulse modulation signal

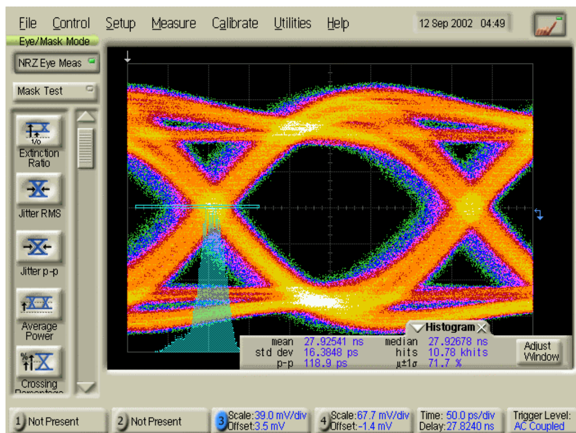

Fig. 3.6: Transmit output with the narrow-pulse-jitter injected

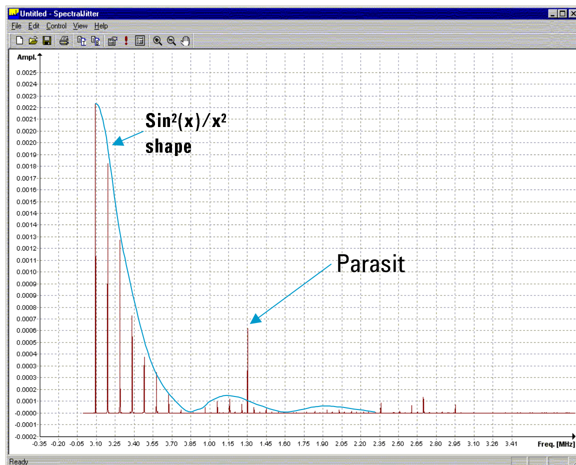

Fig 3.7: Spectral analysis of the injected narrow-pulse-jitter

Another powerful feature of the method is the analysis of wideband "white noise" phase variation transferred from the reference clock input through your chip's PLL into the transmit output signal. Fig. 3.8 shows a random 3.125Gbps high speed data signal output with additional 80MHz wideband random jitter injected into the reference clock. Due to the spectral loop characteristic of the high-speed bit clock synthesizing PLL the "white noise" phase variation gets shaped to "pink noise" phase variation clearly indicating the bandwidth and jitter peaking characteristic of the PLL (see fig. 3.9). Low frequencies in the "white noise" jitter get completely transferred by the PLL because the PLL can track a slowly varying phase. At a certain frequency however, the PLL loop filter first tends to peak a little bit according to the goal of critical loop damping before it finally attenuates all the fast varying phase variations to keep the loop stable. This marks the beginning of the loop filter roll-off or the so-called out-of-band range of the PLL where it can't track the incoming jitter anymore. This measurement also shows that you are also able to assess the spectral properties of your PLLs using this methodology.

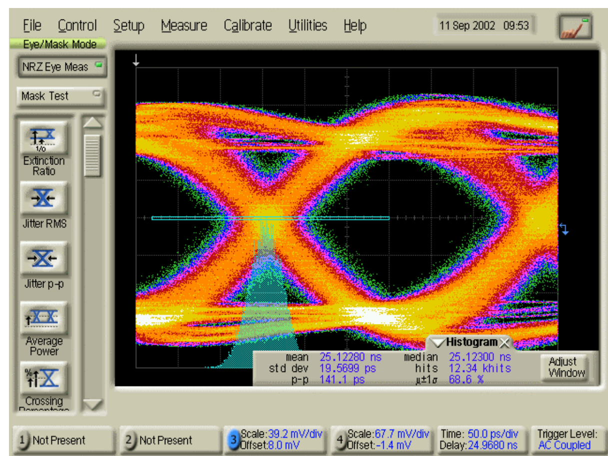

Fig.3.8: Transmit output as a result of high bandwidth random jitter injection

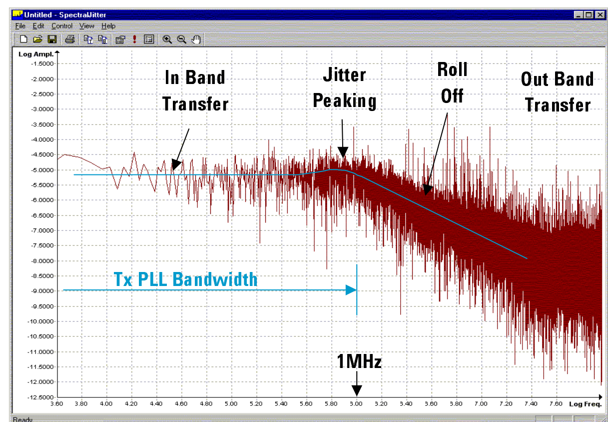

Fig. 3.9: Spectral analysis of the injected random jitter injected

Conclusion

Re-use of your existing at-speed capture-and-compare equipment using the approach of spectral analysis of error densities enables you to do spectral jitter decomposition on a per-port basis. You are not bound to clock-like signals as with phase noise measurements on a single channel spectrum analyzer. You can use realistic data traffic instead. A linear spectral mapping of jitter modulation into the error density domain is possible in the presence of random jitter. In this case the obtained spectrum completely represents the jitter modulation spectrum. The method shows extremely high sensitivity for detecting smallest periodic contents hidden in dominating random jitter. Using wideband "white noise" jitter injection into the reference clock of a PLL the method further allows you to analyze the characteristics of the PLLs in the frequency domain extracting important parameters such as the loop bandwidth. But what is greatest about this method: you'll get it for free, just by doing some clever post-processing on your already existing equipment.

Further Literature

/1/ A. Papoulis: Probability, Random Variables and Stochastic Processes, McGraw Hill 1965

/2/ J. S. Bendat: A. G. Piersol, Random Data, John Wiley&Sons, 1986

/3/ Yi Cai, Bernd Laquai, Kent Luehman: Jitter Testing for Gigabit Serial Communication Transceivers, IEEE Design and Test of Computers, Jan-Feb 2002

/4/ Bernd Laquai, Yi Cai: Testing Gigabit Multilane SerDes Interfaces with Passive Jitter Injection Filters, IEEE Proc. of the International Test Conference 2001

/5/ T.J. Yamaguchi, M. Soma, M. Ishida, H. Musha, L. Marlasie: A New Method for Testing Jitter Tolerance of SerDes Devices Using Sinusoidal Jitter, IEEE Proc. of the International Test Conference 2002

/6/ J. Wilstrup: A Method of Serial Data Jitter Analysis Using One-Shot Time Interval Measurements, IEEE Proc. of the International Test Conference 1998

/7/ National Committee for Information Technology Standardization (NCITS) T11.2/Project 1230-DT "Fibre Channel – Methodologies for Jitter and Signal Quality Specification" Rev. 5.0, February 21, 2002

This information is subject to change without notice.

Printed on recycled paper

© Agilent Technologies, Deutschland GmbH 2003