Novel Test Infrastructure and Methodology Used for Accelerated Bring-Up and In-System Characterization of the Multi-Gigahertz Interfaces on the Cell Processor

P. Yeung, A. Torres, P. Batra
Rambus, Inc
http://www.rambus.com

Abstract

Design-for-test (DFT) techniques are continuously used in designs to help identify defects during silicon manufacturing. However, prior to production, a significant amount of time and effort is needed to bring-up and validate various aspects of the silicon design in the system. In particular, the use of multi-Gigabit I/O signaling for a high I/O count, high-volume product introduces unique test challenges during these two phases of the product life cycle.

In this paper, we shall discuss the test infrastructure and methodologies used to accelerate bring-up and in-system silicon characterization for high-speed mixed-signal I/O. These ideas will lead to a shortened time to market (TTM) at a lower cost. As a case study, we shall illustrate these techniques used in the development of the Rambus FlexIO™ processor bus and XIO™ memory interface used on the first generation Cell processor (aka Cell Broadband Engine™ or Cell BE). Cell was co-developed by Sony Corporation, Sony Computer Entertainment Inc, Toshiba Corporation, and IBM and is used in the Sony PlayStation®3 (PS3™) game console and other intense computational applications. The Cell processor uses 5Gbps links for the processor's FlexIO system interface and 3.2Gbps links for the processor's XDR™ memory interface. This per pin bandwidth translates into a system interface with a bandwidth of 60GB/s and a memory interface with a bandwidth of 25.6GB/s, respectively.

1. Introduction

The typical SOC product development cycle begins with a set of product requirements and concludes with mass production. Three vital steps in this cycle are bring-up, component test and validation, and system test and validation. These efforts also include silicon and system characterization of the I/O subsystem. This paper explores the test features designed into the multi-Gigabit, high-speed interfaces used in the Cell microprocessor. This case study illustrates how some of the I/O design features can be utilized in novel measurement techniques to help accelerate the bring-up and in-system silicon characterization (ISSC) phases of the product lifecycle. An introduction to the Cell processor and its product application can be found in [1].

1.1 Bring-Up and Test Challenges

Historically, I/O bring-up has been a straightforward exercise. The I/O circuits of the past could be simplified as inverters with the expectation they would work on the first attempt to drive or receive a valid signal. However, the multi-Gigabit signaling of today is much more complex. Newer designs such as FlexIO and XIO utilize circuits such as delay-locked loops (DLL), phase-locked loops (PLL), current-mode drivers, pseudo random bit sequence (PRBS) generators and checkers, transmit and/or receive equalizers, periodic timing calibration, loop-back paths, on-die termination, swing calibration, current biasing circuitry and control registers that require software initialization as part of the system initialization sequence. These and other circuit features are a consequence of the increasing design complexity necessary to robustly transfer data at higher data rates which translate into shrinking voltage and timing budgets. In addition to potential circuit related issues at high data rates, the designer must take into consideration the effects of signal and power integrity degradations. Finally, the industry's migration to advanced, deep sub-micron technologies forces designers to measure I/O margin as a function of the manufacturing process as well. It is apparent that testing and validating the high-speed I/O subsystem requires a well conceived bring-up and characterization strategy.

SOC bring-up is typically the first time various subsystems are integrated and is usually when challenging anomalies arise. The various subsystem interdependencies often make it very difficult to isolate problems. For example, if a database program crashes after 24 hours of running on a system, what process should the bring-up team use to root-cause the failure to a component loading issue on the printed circuit board (PCB)? Another complication driven by TTM is the trend for silicon to be

978-3-9810801-2-4/DATE07 © 2007 EDAA

brought up in systems in parallel with the development of manufacturing tests. This complicates problem isolation because there is no easy way to tell whether the inadequately tested silicon has manufacturing issues or the problem is related to design errors in the silicon or in the system.

A major challenge during bring-up is enabling multiple teams to make progress simultaneously rather than requiring each team to perform tasks sequentially. For example, testing the memory subsystem is interdependent with testing the software used in the processor core to generate traffic and vice versa. This interdependency of tasks can slow down the progress of the entire bring-up process. Finally, since bring-up often occurs in multiple geographical locations over many systems, I/O bring-up needs to be done using flexible, portable, and inexpensive tools to allow for remote diagnosis.

1.2 In-System Silicon Characterization (ISSC)

Once the bring-up phase is completed, the ISSC phase starts. The purpose of performing ISSC is to evaluate how the silicon behaves in a realistic system environment. This is differentiated from tester-based silicon characterization in that the focus of tester based characterization is to evaluate the silicon under ideal operating conditions. For example, the multi-Gigabit I/O could behave very differently depending on the channel impedance and on the reference clock that it receives. Often, a significant amount of effort is needed to correlate the results between these two methods.

For designs which will be used in high-volume products it is beneficial to test over the expected range of manufacturing variation. Early in the product life, sample sizes are small. One technique that has been used widely to expedite manufacturing variance analysis is to test simulated worst case system by testing different combinations of silicon processes (P), operating voltages (V) and temperatures (T). The number of PVT test cases would be on the order of hundreds. The testing of this large matrix can consume lots of time and expensive test equipment. Investing time to optimize this stage of the development often helps the TTM and minimizes the cost of the product development.

One of the main difficulties during the ISSC phase is how to quantify the I/O performance. Traditionally, engineers have used oscilloscopes to quantify I/O performance and many probing techniques were developed [5] [6]. Many of today's high performance systems use ball grid array (BGA) and system-in-a-package (SIP) packaging technology. These packaging technologies, coupled with micro-strip PCB routing, effectively eliminate the possibility of using oscilloscope probing methods since probe points are inaccessible.

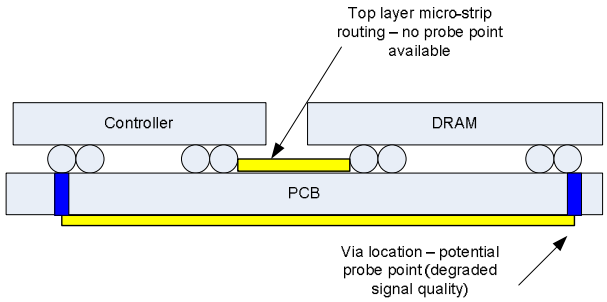

Figure 1. Micro-strip routing between two BGA devices

Refer to Figure 1 for an example of how a high speed memory signal may be routed between an XIO controller and an XDR DRAM. Even in cases where there is a probe point available, probing in the middle of a non-ideal transmission line often does not provide the accuracy required for ISSC.

Another complication is the sheer quantity of multi-Gigahertz signals. In the case of the Cell processor, there are close to 200 pairs of high speed I/O signals. It is not practical to probe every one of them to quantify the I/O performance.

2. Measurement Components

A principal purpose of in-system measurements is to quantify the quality of the I/O signaling over PVT conditions. The components to be measured in the system span three development areas: chip design, board design and software design.

2.1 Chip-Level Features

The FlexIO interface integrates four key features to allow for in-system measurements. These features include FlexPhase™ technology which is a Rambus circuit technology that enables flexible phase relationships between signals, allowing precise on-chip alignment of data and clock, a PRBS generator and checker, a calibration engine and an independent register interface.

All of the FlexIO high-speed receivers use FlexPhase technology to shift the sampling clock with respect to the incoming stream of data. A calibration engine uses this feature to scan through the incoming data stream to locate the center of the valid data region. Once the center is acquired, the phase relationship is recorded and used as a virtual sampling clock. The calibration engine also stores information related to the quality of the eye. The design also contains error and cycle counters for completeness.

In order to determine whether the receiver is receiving the data correctly, the transmitter must send a known sequence and the receiver must have the ability to compare the received bit sequence to the known good sequence. In the FlexIO interface, the transmitter uses a deterministic PRBS and the receiver has the corresponding PRBS checker. The XIO memory interface uses a similar concept

however it uses a pattern buffer to store the PRBS data instead of a PRBS generator since the XDR DRAM and the processor cache are natural storage elements. On XIO, much of the design complexity is on the controller side to make the XDR DRAM simpler to design and manufacture. For simplicity, the FlexIO interface will be used as an example in this paper.

In order to retrieve the results of the calibration engine, a simple serial interface is used to access the registers in the I/O blocks. The serial interface is independent of the processor's core logic and can be accessed at the physical pin, thus eliminating any dependency on the core logic in the system. Test software can also use the serial interface to override and adjust the virtual sampling clock as well as initiate the calibration engine at specific phase offsets. The test software can also adjust channel and circuit parameters such as equalization, swing and receiver gain.

2.2 Board-Level Features

At the board-level the prototype system provides facilities to adjust parameters pertinent to debug and characterization such as process, voltage, and temperature. It also has the facilities to measure current for power calculations, and to monitor clocks and other key signals. Finally, there is also a provision on the board to facilitate frequency adjustment.

Testing over process (i.e. different silicon) variation is still a manual task which involves replacing the microprocessor in the system. The prototype system uses a socket to facilitate device replacement. The socket is also designed to accommodate a thermal unit to isolate and vary the chip temperature. Temperature variation is automatically controlled by the test software.

The prototype system used for this paper intentionally employed a voltage regulator for each of the voltage domains in the microprocessor. The system also provides additional regulators for the other devices in the system, thus isolating the different blocks for system characterization. For production, the voltage regulators can be combined for cost reduction.

Adjustable as opposed to fixed voltage regulators are employed to allow for in-system characterization. Using a digital potentiometer is one approach to in-system voltage adjustments [4]. In our case, the adjustments are accomplished by using current-steering techniques at the regulator's feedback node. The test software has the ability to control and measure the voltage regulator output due to the current-steering circuit design.

Each regulator also uses a high-current, low-impedance sense resistor in the electrical path to its load. A differential sensor is attached to the two resistor terminals to measure the amount of current supplied to the load by the regulator. As with all of the other measurement components in the system, the sensor is sampled by the test software.

2.3 Test Software Features

The test software application helps orchestrate debug and characterization activities in the lab. For this project, the team uses Rambus' LabStation™ software. This infrastructure offers features for a seamless transition from a dynamic, fast-paced debugging environment to a more stable, automated characterization environment. The software infrastructure offers an interactive graphical user interface (GUI) to facilitate impromptu lab experiments. The software infrastructure also offers a test-oriented scripting language to facilitate automated test execution. Through either interface, the user is able to control the system at any level, i.e. chip, board, and system including external test equipment.

Controlling the system begins at the chip-level registers. LabStation software creates an abstract yet intuitive GUI to represent the registers that reside within the chip under test. The registers are presented in the GUI in a hierarchical manner. The registers are also referenced by the scripting language in the same hierarchical manner. Therefore, performing low-level register operations in the GUI can be mimicked or recorded in the native scripting language. The scripts themselves may be used across debug platforms or employed in the characterization environment. The same hierarchical concept is used to control board-level devices, system-level devices and off-the-shelf test equipment.

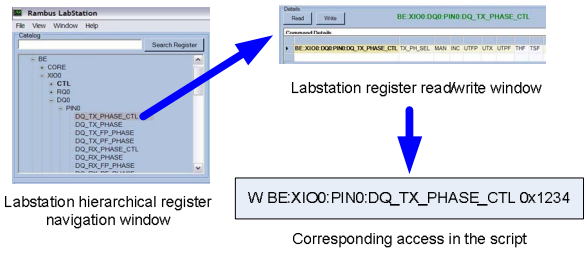

Figure 2. LabStation GUI and scripting example

Although the registers are treated as logical paths, they behave as variables would in any standard programming language. Variables in the scripting language are updated by read/write operations. Since register operations are introduced in the scripting language, it is easy to wrap a series of register operations within control loops. Likewise, board-level, system-level and external devices such as thermal controllers and other test instrumentation could be treated in the same manner. Therefore, control of the devices in the measurement system is simplified to basic read/write operations which are analogous to read/write operations to the registers in the micro-controller.

The test or design engineer can focus on producing simpler test routines since all of the measurement devices are addressed in a similar manner. Most test routines are simplified to a set of precise operations wrapped in a series of nested loops. The following pseudo code is used to

measure the timing margin of the XDR memory channel across processor and DRAM voltages and temperatures.

```
for processor temperature = min,typ,max
  for XDR temperature = min,typ,max
    for processor core voltage = min,typ,max
      for processor I/O voltages = min,typ, max
        for XDR voltages = min,typ,max
        {
          flexphase read timing for all pins;
          flexphase write timing for all pins;
        }
```

Figure 3. Timing margin pseudo-code over VT conditions

The LabStation software also has the ability to make system calls to other software to post-process the output. Thus, with the click of a button, the engineer is able to execute the test script, monitor the output, collect and post-process the test results.

3. Assembling the Measurement System

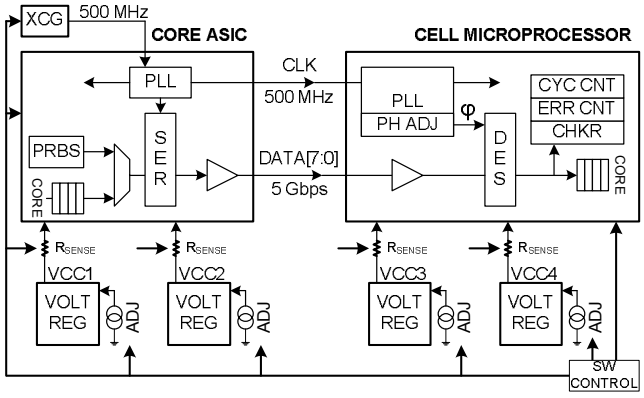

Figure 4. Assembled Measurement System for First Generation Systems Using FlexIO Technology

The measurement components discussed in section 2 are assembled as shown in the preceding diagram. It is important to note that the measurement instruments are embedded in the prototype system. This reduces the dependency on external test instruments for a lab bench environment. This also makes it feasible to replicate numerous measurement stations at a lower cost.

3.1 Hardware

As shown in the figure above, the measurement elements required to quantify the quality of the high-speed signaling are embedded in the silicon design. All of the other measurement components provide the means to vary the environmental conditions. These components are integrated at the board or system level with the exception of the thermal heating and cooling unit. If necessary, test instruments that utilize the general purpose interface bus (GPIB) protocol may also be used to supplement the measurement system.

3.2 Software

Complete control of the system is possible because all of the components can be addressed by a single programming application, LabStation software. In Figure 1, LabStation software controls and monitors the chip level registers, the voltage regulators, the current sensors, and the clock frequency. The software tool also controls the external thermal unit that connects to the chip socket. If necessary, it is also possible to control external test instruments through a GPIB interface. The external instruments also appear in the GUI interface and may thus be addressed in the native scripting language as well.

LabStation software has been proven to facilitate both bring-up and characterization efforts, however manufacturing test can also benefit from tools like LabStation software. The test software may also be invoked as a dynamically-linked library to receive commands from third-party software applications or test environments. This implies that test engineers do not have to go through the exercise of creating device drivers for the measurement components found in a system like the one shown in Figure 4.The third-party application(s) may invoke LabStation software to serve as a device driver to control and monitor devices in the measurement system. This also means that the diagnostic or specific test scripts that were developed during the phases of bring-up and characterization can be reused in manufacturing as well.

4. Debug and Measurement Techniques

This section begins with a basic overview of the Cell microprocessor's initialization sequence pertaining to the high-speed interfaces. After this brief introduction, the paper presents some basic diagnostic techniques that are typical of the debugging process. The discussion then proceeds to more advanced techniques to address the challenges of measuring system level margin.

4.1 Overview

As part of the normal boot-up procedure, the FlexIO high-speed interface is initialized by transmitting PRBS patterns on all transmission lines and launching the corresponding calibration engines. A calibration engine utilizes a PRBS checker and the ability to tune the timing of the virtual data samplers, i.e. FlexPhase technology The results of the calibration engine may be used to quantify the amount of margin on the high-speed channel. These results are stored on a per pin basis thus allowing software to extract the system margin over the entire high-speed interface. Using a framing pattern, the individual I/O bits are aligned with respect to each other. At this point, the processor can begin to transmit packets of information with embedded cyclic redundancy check (CRC) information. These packets are the product of intercommunication between the core devices as real-time microprocessor applications are executed.

To utilize the aforementioned calibration abilities for debug and characterization, it is important to understand how the calibration engines quantify the valid data region. For a given bit time or unit interval (UI), there are three distinct regions which are typical of any eye diagram: the left failing region, the center passing region and the right failing region. The quality of the received signal can be measured by the size of the center passing region as a proportion of the overall bit time. Larger passing regions help meet or exceed the target bit error rate (BER) for a given link.

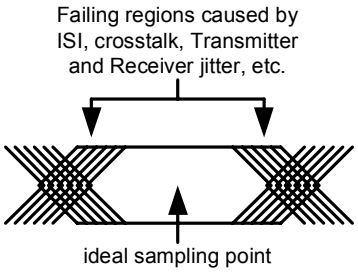

Figure 5. Sample Eye Diagram Over 1 UI

The phase calibration process is an automatic operation that occurs on the order of microseconds. The calibration engine sets the virtual data samplers to a zero phase offset. The PRBS checker is first cleared and then is used to check the incoming PRBS pattern over a programmable number of over-samples. Error and cycle counters can be optionally employed to keep track of bit error rates if desired. The sampler is then adjusted by one phase unit and the checking process is repeated. At the end of this procedure, the calibration engine leaves markers to denote the fail-to-pass and pass-to-fail boundaries based on the largest contiguous passing region that was detected. The samplers are subsequently updated with the phase adjustment value that corresponds to the center of the largest passing region. The calibration sweep occurs over 128 phase steps which are equally distributed over one unit interval. For example, at 5Gbps (200 Pico-seconds UI) the phase step resolution is 1.56 picoseconds.

In summary, the calibration engine and FlexPhase circuitry are integral parts of the I/O circuitry. These features are used to calibrate the I/O subsystem of the Cell microprocessor. Furthermore, test software can benefit from these embedded mechanisms to facilitate debug and characterization as well.

Test software, such as the LabStation software, has the ability to override the phase adjustment register or simply reinitiate the calibration engine to perform basic debugging functions in the lab. For simplification, FlexPhase technology shall represent the phase adjustment capability, the calibration engine and the PRBS checker from this point forth. There are several tests that can be accomplished in the lab with the FlexPhase feature. These tests include but are not limited to: connectivity testing, measuring I/O timing margin over PVT conditions, jitter and BER testing, diagnostics, and silicon parameters such as receiver sensitivity and transmitter swing linearity.

4.2 Connectivity Testing

The connectivity test is a fundamental test that is desired during bring-up, test and board manufacturing. The FlexPhase feature is utilized to quickly assess the state of the system's connectivity, especially in early prototypes when the manufacturing facility is not prepared to screen the printed circuit assemblies with tests fixtures such as an In-Circuit-Test (ICT) fixture. Even if ICT fixtures are available, early prototypes rely heavily on sockets which can often lead to connectivity issues.

Connectivity tests are performed at speed which is an added benefit of the FlexPhase circuitry. Finding a marginal I/O link that does not meet a minimum UI threshold is a trivial task and can be easily and expeditiously identified. This is especially useful when the system's communication packets report CRC errors. The following diagram is an example of the UI measurements for 32 I/O links over different types of system activity. The measured UI values demonstrate that the entire interface exceeds a UI threshold which denotes that there is enough signaling margin. To complement the FlexPhase connectivity testing on data pins, the XDR memory interface also has dedicated memory address and request bus connectivity testing between the controller and each of the XDR DRAMs.

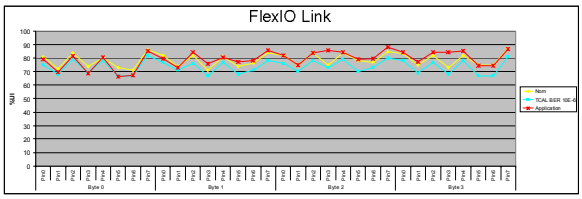

Figure 6. Sample UI Plots for Initial Calibration with PRBS7, Initial Calibration with PRBS15 and Running a Test Application at 5Gbps

4.3 Measuring Timing Margin Over Conditions

Aside from connectivity testing, more comprehensive testing can be accomplished with the FlexPhase circuitry. The designer has the ability to extract the information needed to assess the signal quality (UI) under specific environmental conditions. The test software (LabStation software) has the ability to control both on-board and external test and measurement instruments. Therefore, the designer or test engineer can construct plots to report the signal quality over conditions such as process, voltage, frequency, skewed impedance packages, skewed impedance boards, PLL settings, temperature, etc.

As stated in the introduction, these factors are important to measure for high-volume manufacturing where relatively large tolerances in the design constraints are allowed with the desire to reduce the system's manufacturing costs. It is also worth noting that the cost in time and resources to characterize the high-speed signals is minimized since expensive oscilloscopes and BER testers are not employed to measure the signal quality. Using such instruments is also limited by the number of measurement channels, thus limiting the measurements to only a small percentage of the overall I/O interface.

4.4 Jitter and Bit Error Rate Testing

The error and cycle counters that are embedded in the FlexIO design can be utilized to produce jitter terms (RJ and DJ) along with BER plots, i.e. bathtub curves [3]. BER measurements are taken by initiating the calibration engine. However, the error count and corresponding cycle count is extracted from the system as the phase adjustments are performed. BER bathtub plots can then be generated as exemplified in the following diagram.

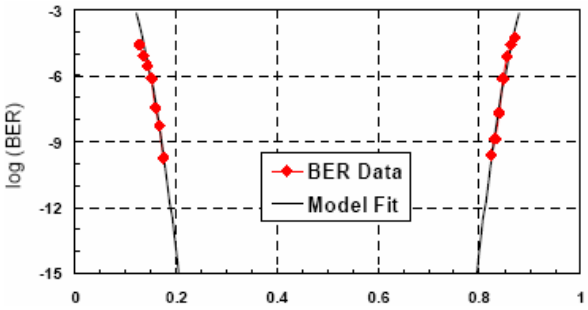

Figure 7. Measured System Jitter and Double Delta Model Fit at 6.4Gbps (DJ=23.3ps, RJ=2.62ps)

4.5 Measuring I/O Design Parameters

There are several chip-level parameters pertaining to the I/O links that can also be measured in the system. For example, the FlexIO cell also has the ability to adjust the transmitter output swing from 0 mV to well beyond the swing of 250 mV, single-ended. Using the calibration results, which include TX, RX and the channel, the test engineer has the ability to measure the receiver sensitivity in the system. Figure 8 is an example of the data plots for swing and receiver sensitivity in the system.

4.6 Remote Diagnostics

Finally, all of the aforementioned measurements and tests may be performed in the system by executing the specific test scripts. This facilitates remote debugging and testing of systems deployed in the field.

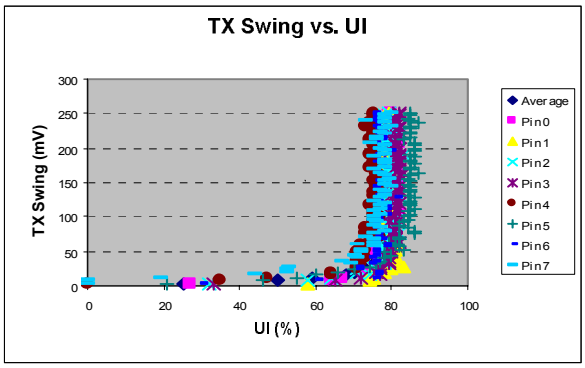

Figure 8. Transmitter swing vs. UI plot

5. Conclusion

Multi-Gigahertz I/O circuit designs are progressively more complex in order to address aggressive timing budgets. The same circuit features to compensate for non-ideal circuit and channel characteristics, if enhanced with test in mind, can be used to accelerate debugging and characterization efforts. These enhanced features can complement the traditional measurement techniques to meet the challenges of quantifying the I/O and system-level margin. However, to effectively use these I/O circuit features for test purposes, the designer must architect an appropriate test hardware and software infrastructure early in the design stage.

6. Acknowledgements

We extend our thanks and gratitude to Wai-Yeung Yip, David Nguyen, Kenyon Han, Melissa Frank, Paula Tostado and Keisuke Saito of Rambus Inc. for reviewing the paper.

7. References

[1] Cell Broadband Engine Architecture Version 1.0, April 8, 2005

[2] K. Chang, et. Al. "Clocking and Circuit Design for a Parallel I/O on a First-Generation CELL Processor", *ISSCC* 2005, pp 526-52

[3] C. Madden, et. Al. "System-Level BER Test and Jitter Extraction of a 6.4Gbps Parallel Chip to Chip Bus on the First Generation CELL™ Processor", *Proceeding of IEEE EPEP*, 2005.

[4] Maxim Integrated Products, Inc. "Step-Up DC-DC Converter Calibration and Adjustment Using a Digital Potentiometer", Application Note 226

[5] Agilent, Inc. "Improving Usability and Performance in High-Bandwidth Active Oscilloscope Probes" Application Note 1419-02

[6] Agilent, Inc. "Optimizing Oscilloscope Measurement Accuracy on High-Performance Systems with Agilent Active Probes", Application Note 1385