As originally published in the SMTA Proceedings

# Creating Reusable Manufacturing Tests for High-Speed I/O with Synthetic Instruments

**Louis Y. Ungar, Neil G. Jacobson, T.M. Mak,**
A.T.E. Solutions, Inc.
El Segundo and San Jose, CA

**Abstract**

There is a compelling need for functional testing of high-speed input/output signals on circuit boards ranging from 1 gigabit per second(Gbps) to several hundred Gbps. While manufacturing tests such as Automatic Optical Inspection (AOI) and In-Circuit Test (ICT) are useful in identifying catastrophic defects, most high-speed signals require more scrutiny for failure modes that arise due to high-speed conditions, such as jitter. Functional ATE is seldom fast enough to measure high-speed signals and interpret results automatically. Additionally, to measure these adverse effects it is necessary to have the tester connections very close to the unit under test (UUT) as lead wires connecting the instruments can distort the signal. The solution we describe here involves the use of a field programmable gate array (FPGA) to implement the test instrument called a *synthetic instrument (SI)*. SIs can be designed using VHDL or Verilog descriptions and "synthesized" into an FPGA. A variety of general-purpose instruments, such as signal generators, voltmeters, waveform analyzers can thus be synthesized, but the FPGA approach need not be limited to instruments with traditional instrument equivalents. Rather, more complex and peculiar test functions that pertain to high-speed I/O applications, such as bit error rate tests, SerDes tests, even USB 3.0 (running at 5 Gbps) protocol tests can be programmed and synthesized within an FPGA. By using specific-purpose test mechanisms for high-speed I/O the test engineer can reduce test development time. The synthetic instruments as well as the tests themselves can find applications in several UUTs. In some cases, the same test can be reused without any alteration. For example, a USB 3.0 bus is ubiquitous, and a test aimed at fault detection and diagnoses can be used as part of the test of any UUT that uses this bus. Additionally, parts of the test set may be reused for testing another high-speed I/O. It is reasonable to utilize some of the test routines used in a USB 3.0 test, in the development of a USB 3.1 (running at 10 Gbps), even if the latter has substantial differences in protocol. Many of the SI developed for one protocol can be reused as is, while other SIs may need to undergo modifications before reuse. The modifications will likely take less time and effort than starting from scratch. This paper illustrates an example of high-speed I/O testing, generalizes failure modes that are likely to occur in high-speed I/O, and offers a strategy for testing them with SIs within FPGAs. This strategy offers several advantages besides reusability, including tester proximity to the UUT, test modularization, standardization approaching an ATE-agnostic test development process, overcoming physical limitations of general-purpose test instruments, and utilization of specific-purpose test instruments. Additionally, test instrument obsolescence can be overcome by upgrading to ever-faster and larger FPGAs without losing any previously developed design effort. With SIs and tests scalable and upward compatible, the test engineer need not start test development for high-speed I/O from scratch, which will substantially reduce time and effort.

**Introduction**

Today's circuits operating at multi-Gbps speeds have complex failure modes for which tests must be developed. Design verification tests, utilizing stand-alone oscilloscopes and Bit Error Rate Testers (BERT) can identify failures due to design errors, omissions or variations. Once the design is corrected, it is no longer feasible to use these long tests or discrete instruments with long set-up times to test each production unit. The manufacturing tests need to be automated to handle the large volume and tests must be oriented to the failure modes they expect to exhibit, specifically those caused by faulty circuit elements and assembly processes.

The term "high-speed" is subjective, so we will limit our discussion in this paper to high-speed I/O buses that are listed and described in Table 1.

**Table 1–Typical High Speed I/O Buses**

| High Speed I/O Buses | Purpose | Speed | Features |
|---|---|---|---|
| **USB 3.0** | External peripheral interface standard for Plug-and-Play communication between a computer and external peripherals over a cable using bi-directional serial transmission | 5 Gbps | Also provides power for plugged in peripherals and charging |
| **USB 3.1** | Same as USB 3.0 | 10 Gbps | Same as USB 3.0 |
| **SATA** | Serial ATA is a computer bus interface that connects host bus adapters to mass storage devices such as hard disk drives, optical drives, and solid-state drives. | 1.5 to 3.0 Gbps, with extensions to 6.0 and 16.0 Gbps | It uses two differential ports, A and B. eSATA, uses a more robust connector, longer shielded cables, and stricter (but backward-compatible) electrical standards. |
| **PCIe** | High-speed serial computer expansion bus standard, designed to replace the older PCI, PCI-X, and AGP bus standards. | 2.5, 5, 8 or 16 Gbit/s. | Transmit and receive are separate differential pairs, for a total of four data wires per lane. |
| **HDMI** | Has 3 functions: • Transition Minimized Differential Signaling (TMDS) for high speed video interface. • Display Data Channel for low speed $I^2C$ bus for configuration. • Optional Consumer Electronics Control (CEC), a low speed interface for remote devices. | Up to 18 Gbit/s (2.25 GBytes/s) | An HDMI connection can either be single-link (type A/C/D) or dual-link (type B) and can have a video pixel rate of 25 MHz to 340 MHz (for a single-link connection) or 25 MHz to 680 MHz (for a dual-link connection). |
| **InfiniBand (IB)** | Computer-networking communications standard used in high-performance computing that features very high throughput and very low latency. It is used for data interconnect both among and within computers. | Single data rate up to 24 Gbps. Exceeds 290 Gbps for enhanced data rate (EDR). | Also utilized as either a direct, or switched interconnect between servers and storage systems, as well as an interconnect between storage systems |
| **1-Gb and 10-Gb Ethernet** | Various technologies for transmitting Ethernet frames at rates above 1 Gbps. | 1 Gbps and 10 Gbps | 10 Gigabit Ethernet defines only full duplex point-to-point links which are generally connected by network switches; shared-medium CSMA/CD operation has not been carried over from the previous generations Ethernet standards. |
| **10 (X) Attachment Unit Interface (XAUI)** | A standard for extending the XGMII (10 Gigabit Media Independent Interface) between the MAC and PHY layer of 10 Gigabit Ethernet (10GbE). | 10 Gbps | Provides physical separation between MAC and PHY components in a 10 Gigabit Ethernet system distributed across a circuit board. |

For low-speed circuits or those exhibiting catastrophic failure modes the traditional approaches have been sufficiently successful. Inspection systems – both optical and X-ray - have done an effective job in locating visible defects, which can comprise more than 80% of the defects. Electrical faults, however are elusive to inspection and have been relegated to manufacturing defects analyzers (MDAs) and in-circuit testers (ICTs) using both permanent and flying probe test access methods. These automatic test equipment (ATE) types are tasked with finding the remaining 20% or so of the faults. How effective ATE is in finding all the remaining faults has been a subject of controversy for years. It makes sense to add functional board test (FBT) to the test mix so that functional failures can be identified at the board level. In high volume

production, the functional test should be performed by a functional ATE, which often needs to be custom designed with appropriate instrumentation. [1] Moreover, test program development is costly due to the need for engineering expertise time and due to time-to-market penalties. Faults that escape ICT must be detected and identified by FBT. Typically, FBT tests the UUT at the speed of the FBT, which tends to be much slower than that of the UUT. Thus, high-speed signals are not properly tested in an automated way. If the failure is detected at the system level – or worse – by the customer, it is much more expensive to mitigate the problem. Detecting high-speed I/O faults *after* FBT can result in diagnostic ambiguity. Without knowing the root cause, boards or even modules are often thrown away rather than repaired. In high volume manufacturing, this can be quite wasteful and detrimental to the environment, and if the failures escape to end users the company's reputation will suffer.

There are several reasons why FBT is slower than the UUT. The routing of high-speed signals through even a few inches of cabling can introduce significant signal delays, noise and attenuation that will compromise stimuli and measurements. For instruments to be effective, tests need to meet test accuracy ratios (TARs) of about 10, meaning that the instrument must be 10 times as accurate as the tolerance of the UUT. Instruments that meet such requirements for even a 5 Gbps signal are expensive and may not be practical for a manufacturing test environment, especially when volume dictates several test stations. Controlling the instruments through test programs adds another layer of complication. In all cases, the test engineer will be responsible for developing all tests from scratch, which is a time-consuming and complex process. Even when the tests detect high-speed failures, the test engineer needs to create more complex diagnostic tests that can effectively diagnose the root cause of the failure and provide specific repair instructions. This creates a bottleneck that our approach aims to overcome.

In this paper, we will describe a novel approach to deal with the high-speed test issue. Utilizing a field programmable gate array (FPGA) for performing the high-speed I/O tests in conjunction with the FBT can offer several advantages:

1. The FPGA-based high-speed tester (FPGA-HST) can be placed physically close to the high-speed I/O of the UUT, minimizing attenuation or interference from the use of longer cables and probes.
2. The FPGA-HST can be developed targeting a specific high-speed I/O bus, e.g. USB 3.0, USB 3.1, HDMI, SATA, Gigabit Ethernet (GbE), etc. While it is not trivial to develop such tests, once developed, the tests are reusable for UUTs that share the same bus structure.
3. In developing the test sets, the FPGA-HST can utilize structures to implement instrument functions, which we call *synthetic instruments (SIs).* SIs can be as basic as a voltmeter or a BERT, more complex like a Serializer/De-serializer (SerDes) tester, or even as complex as a SI capable of emulating a USB 3.0 at various layers, such as the physical (PHY) layer, the data link layer and even the network layer in an Open Systems Interconnection(OSI) model [ISO/IEC 7498-1]. Using SIs as components of the FPGA-HST allows the reuse of tests even across different high-speed I/O buses.The I/O bus test can be reused for any UUT with the same bus.
4. The test engineer can more readily develop tests using a common test programming language to operate the instruments. Some IEEE standards and standard test languages exist that can be used to describe complex test signals from basic signals and from other more complex signals. In this manner, ever-more complex signals can be generated. SIs can be created that produce or evaluate complex signals, thereby making the FPGA-HST capable of producing signals that are not likely to exist in traditional instruments. Tests can arrange/re-arrange combinations of SIs – simple and complex – to test and provide diagnoses. The test engineer can generally achieve higher test coverage and reduced test program development time with a more complex SI.
5. The tests themselves become both scalable and reusable. For example, a SerDes test used for one high-speed bus can be retained and reused for a different SerDes high-speed bus. Complete bus tests or *test sets* can be reused for many different UUTs. Even complete test sets can be modified in the process of creating test sets for other UUTs. For example, parts of a USB 3.0 test set can be reused as an SI in the development of certain parts of a USB 3.1 test set, thereby reducing test program development efforts substantially. (This is helpful even if the USB 3.0 test set cannot be considered a proper subset of a USB 3.1 test).
6. A variety of SIs can be used to perform tests that are not feasible with traditional ATEs. For example, on an 8-bit wide parallel high-speed I/O bus, the signals can be connected to 8 different voltmeter (synthetic) instruments. and measurements taken simultaneously. Notwithstanding delays and skewing between instruments, (which can be measured between the instruments and taken into consideration) simultaneous measurements using multiple instruments can assist with some tests. Traditional ATE typically uses a single voltmeter and switching mechanism, which in this example would result in 8 tandem measurements and no information about the relationship of one measurement to any other.

We will describe some of the considerations this new paradigm in testing raises. We will first introduce test terminology used in test standards. We then describe FPGAs and discuss how they can be used as instruments and SIs. We introduce high-speed I/O considerations considering high-speed failure modes and detail the tests that need to be created. An important goal of the test will be to not only detect but also to diagnose failure modes. Our novel approach also allows for testing of link-layer and network (protocol)-layer interactions, unlike most ATE tests that are limited to testing only the physical-layer. We describe the tests we perform on high-speed I/O and explain how the SI within the FPGA serves to realize these tests.

**Test Standards, Signals and Languages**

Attempts have been made by both commercial and military organizations to standardize the way we describe tests and to form consensus throughout the profession. Military and avionics applications use the Abbreviated Test Language for All Systems (ATLAS) [2], and commercial test systems utilize graphical test languages from private firms [3].

It is helpful to clarify some basic terminology we rely in this paper. [4]

- *Tests* are intended to draw distinctions between UUTs that produce expected responses and those that do not.
- To develop a test, one needs to first produce a *test requirement,* which describe the test conditions that should be applied to and collected from a UUT.
- Next, a *test specification* is needed that provides the definition of the tests to be performed on the UUT, with or without fault diagnostics, and without reference to any specific test. It is important to note that a test specification is not the same as a performance specification, though the latter is often used as a reference and a guide by the test engineer developing the test specification.
- The *test procedure* describes the tests, test methods, and test sequences to be performed on the UUT to verify conformance with its test specification, with or without fault diagnostics, and without reference to specific test equipment.
- The *test program* is an implementation of the tests, test methods, and test sequences to be performed and configured for execution on a specific test system.
- In military and avionics applications the test program is combined with the ATE-to-UUT interface and the documentation and that collection of these items is called a *test program set (TPS)*.
- A *test set* is a collection of tests that come together to test an entire module or system. Examples of test sets include testing an entire avionics box. We can also consider a USB 3.0 test to be a test set, since it can test all USB 3.0 connections in a variety of UUTs.

Tests use *instrument*s that apply and/or collect responses. Instruments produce and/or evaluate well defined and specified signals, often called *test signals*. Instruments can be described in various test languages. For example, the IEEE-1687 [5] uses an Instrument Connectivity Language (ICL) to describe the interface between an instrument and the UUT it is testing. It also details a Procedural Description Language (PDL) that describes the test procedure. IEEE-1641 [4] provides for a set of Basic Signal Component (BSC) from which more complex, Test Signal Framework (TSF) constructs can be built. BSCs and TSFs can be expressed in Automatic Test Markup Language (ATML), which is an eXtensible Markup Language (XML) as standardized by IEEE-1671. [6] IEEE-1671.2 standardizes the description of instrument models. [7]

In this paper, we suggest using instruments synthesized within the FPGA to realize the instruments and execute test procedures to test high-speed I/O.

**FPGA Configurations Dynamic Reconfiguration**

Field Programmable Gate Arrays (FPGA) are specialty semiconductors that consist of collections of gates and routing channels along with specialized logic blocks that are connected to one another via programmable switch elements. By choosing specific connections between these gates, routing channels and specialized logic blocks using the programmable switch elements, one can synthesize specific electronic logic functions. The programming pattern that describes the actual programmable switch settings is typically stored external to the FPGA in a programmable read only memory (PROM) or generic Flash memory device. The pattern is usually loaded into the FPGA from the memory using the FPGA's JTAG port or a special purpose programming port on the FPGA itself.

The current generation of FPGAs consist of specialized logic blocks that include analog to digital converters (ADC), multipliers, high speed differential drivers and other similar complex logic entities.

Using this combination of circuitry, one can design test instruments on an FPGA. These instruments can include useful test functions such as voltmeters, pattern generators and bit error rate testers. Such instruments can be parameterized and run under the control of a microprocessor attached to or fabricated on the FPGA. One could also use fully functional IP blocks as test instruments. For instance, an IP block that realizes the USB 3.X protocol can be modified to inject known fault patterns and behaviors to validate the operation of an attached USB 3.X peripheral.

The control and run time parameters of the instruments can be represented within the test description language. The FPGA-implemented test instruments that we refer to as synthetic instruments will be described in greater detail in the next section.

Because FPGAs are reprogrammable, the content and composition of the synthetic instruments can be changed in accordance with the test requirements. All FPGAs allow their contents to reprogrammed at power up, allowing the number and type of synthetic instrument to be selected initially based on the tests to be performed. More sophisticated FPGAs also allow for the use of partial reconfiguration. This is a design methodology that allows the test developer to swap in and out functionality from the FPGA at run time. This means that the FPGA would not need to be repowered to change the synthetic instrument selection used for a specific test segment, but rather the instrumentation can be changed on the fly according to the specific test need. This flexibility permits more efficient use of FPGA resources, which can result in the development of more comprehensive and sophisticated tests.

**Synthetic Instruments**

A *virtual instrument* comes about from customizing software and one or more hardware instruments to create a user-defined stimulus and/or measurement system. S*ynthetic instruments (SIs)* are more restrictive but can be considered subsets of virtual instruments. The term *synthetic instrument*s appropriate to use for FPGAs, since the instrument is realized by having an instrument design undergo synthesis (i.e. implementation within an FPGA). However, the term predates FPGAs and even virtual instruments. [8] SIs are detailed in [9], in which a *synthetic measurement system(SMS)* is described as a structure shown in Figure 1. The Stimulus Generator and Response Collector SIs are instruments that apply stimuli to and collect responses from the UUT, respectively. The Controller element can be any computer. The Codec is a digital-to-analog (D/A) or analog-to-digital (A/D) converter. The Conditioner is the signal part of an instrument that connects with the UUT for both stimulus generation and response collection. All three parts of the instrument can be implemented within an FPGA, but the Controller is often contained within an external computer or microprocessor. Many FPGAs today include A/D converters with which to implement the Codec function.

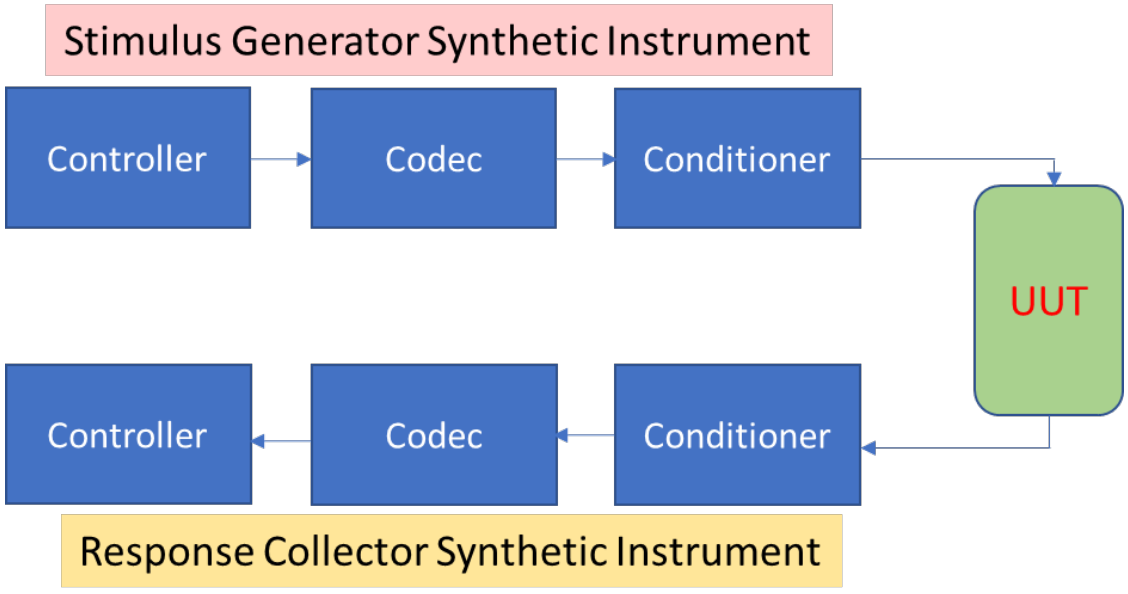


**Figure 1 –Synthetic Instruments (SIs) in a Synthetic Measurement System (SMS) [9]**

Because SIs are abstractions, they can be reconfigured under test program control, the process is called *dynamic reconfiguration*. Because SI functions can change, for example, from a voltmeter to a power meter to a waveform analyzer and even to a pulse generator while still connected to the same UUT pin, SIs can be a more powerful test tool. Additionally, SIs do not need to be the equivalent of hardware instruments. They need not be general purpose, such as a sinusoidal

waveform generator but can be more complex and purposed, such as a generator of complex bus protocols. For example, one can build a SI that mimics a SerDes CDR or even a USB 3.0 protocol.

Figure 2 shows the possible design of a Synthetic ATE. Hardware test instruments that are normally placed between the Controller and the Switch Matrix are now FPGA configured instruments.

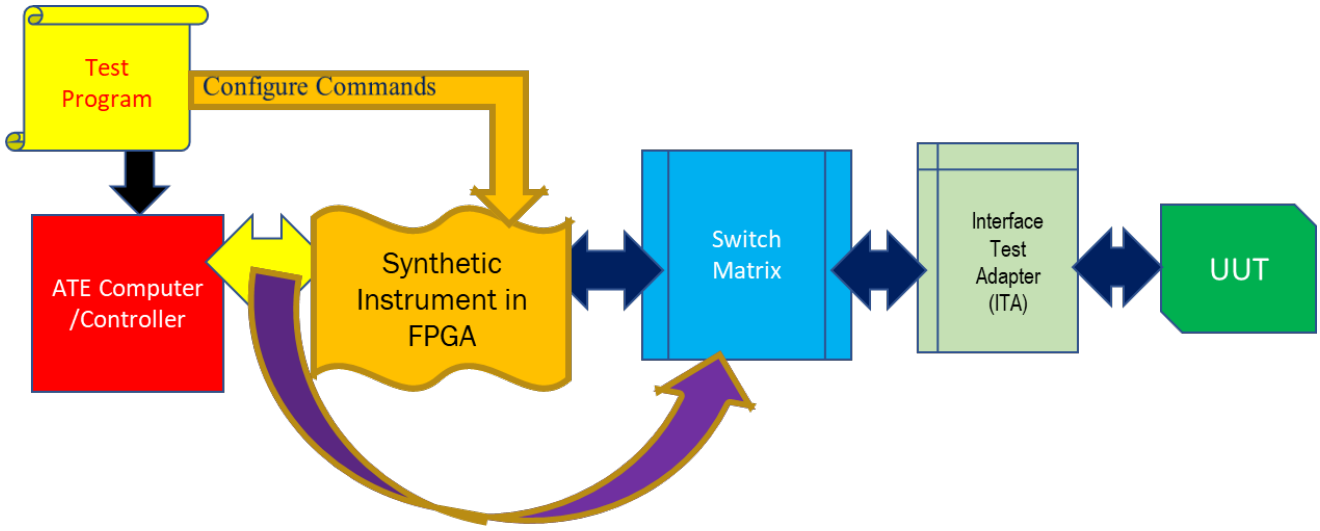


**Figure 2–A Synthetic ATE consisting of a Synthetic Measurement System and Synthetic Instruments**

We can utilize the building block approach introduced in the IEEE-1641 [4] that enables basic signals to become more complex ones by reusing and combining previously defined signals. We can adapt this methodology to building ever-more complex and capable SIs. With industry-wide cooperation, such SIs can be rapidly developed and checked, creating a library of instruments that can test (not only) high-speed I/O buses.

**High-Speed I/O Testing**

In its simplest form, high-speed I/O consists of a pair of differential transceivers - Tx is the Transmitter and Rx is the Receiver. A UUT with high-speed I/O, as shown on the left half of Figure 3 may contain signals either by design for function or design for testability (DFT) through which low-speed (i.e. relative to the high-speed and ATE-compatible) access is available. The ATE on the right-hand side of Figure 3 is the FBT we described earlier. It contains instruments that normally connect to the ATE through standard instrument buses. Some of the instruments could access the Tx and Rx line, but they probably would be impeded by a switching subsystem that would further compromise the true signal on the UUT.

The UUT with the high-speed I/O contains two important blocks. The Parallel-Serial block feeding the Tx converts parallel bus lines to high-speed serial. In high-speed applications, such as SerDes, the transmission also embeds the clock in the serial stream. On the receiving side, a Clock and Data Recovery (CDR) mechanism enables the extraction and reconstruction of data and clock, typically using a phase locked loop (PLL) or other synchronization mechanisms.

In Figure 3 we show a module that is placed between the UUT and the ATE. We intend to implement the functions of this module with an FPGA, that we are calling FPGA-HST in this article. In the Tx test mode, the Pattern Generator applies both deterministic and random test patterns to the UUT through its low-speed access ports. The UUT then serializes it and propagates it through its own Tx.

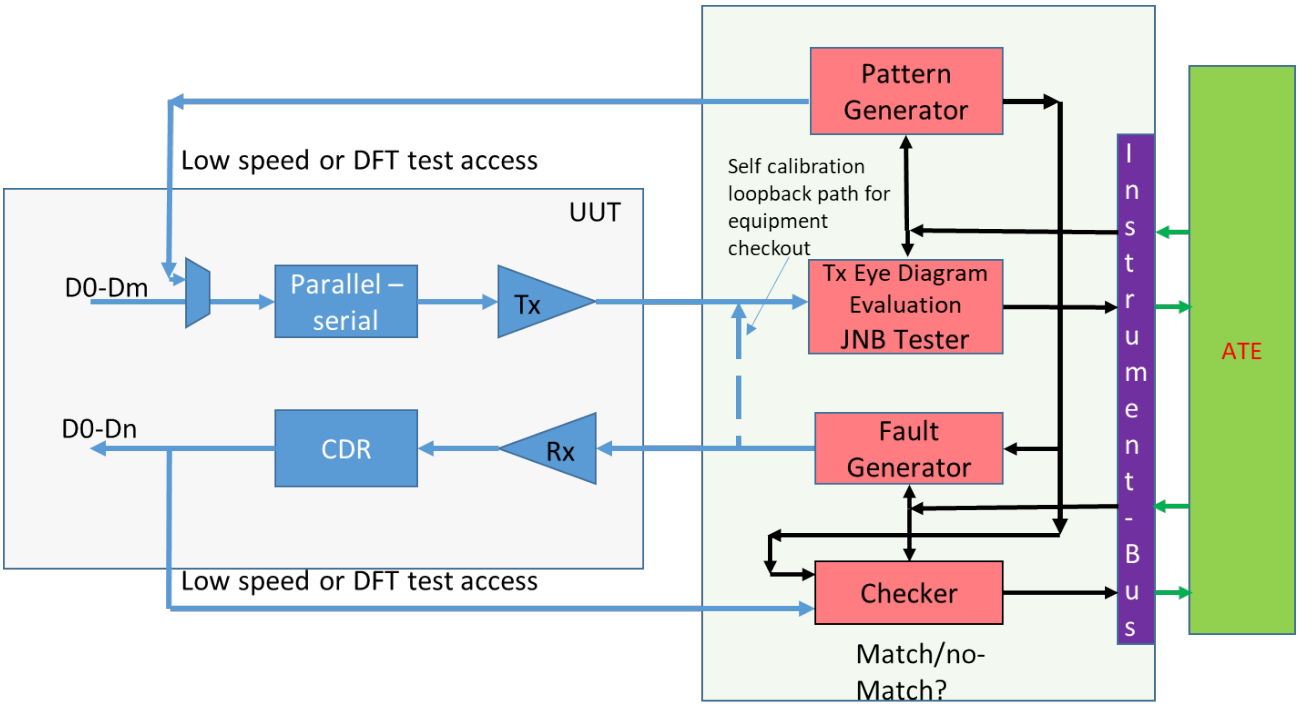


**Figure 3 –Architecture of a FPGA-based High-Speed Tester (FPGA-HTS)**

The Eye Diagram Evaluation mechanism evaluates the quality of the high-speed signal. Figure 4 shows an Eye Diagram and interprets some of the characteristics we look for in the waveform. [10] The Eye Diagram Evaluator can characterize the Tx signal to provide valuable diagnostic information. Analyses of this signal stream also allow jitter, noise and BER (bit error rate) information to be extracted as indicated in Figure 4. In the Rx test mode, the Pattern Generator feeds the fault generator, which provides a jittery signal stream (analog faults) to the UUT's Rx. A properly functioning Rx should be able to reconstruct the data through its various compensation and synchronization circuits. The reconstructed data is then fed back to the FPGA-HST unit for comparison using the Checker circuit block. With UUTs that do not have these low-speed or DFT access ports, we need to loop the UUT's Rx received data back to its own Tx. This will still find faulty communications, but without the low speed or DFT access, some of the diagnostic information is lost and we will be unable to differentiate between Rx and Tx faults.

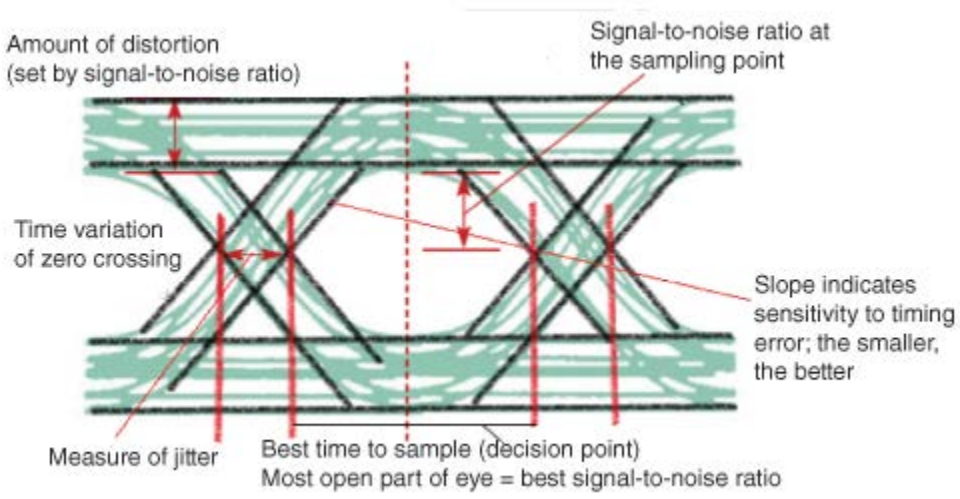


**Figure 4 –An Eye Diagram and its Characteristics [10]**

To test and better diagnose the UUT's Rx, it will be helpful to inject (analog) signals. The Fault Generator block in Figure 3 provides that capability. Fault injection can emulate the workings of neighboring modules within a system environment where the link layer and network or protocol layers are being used. The Fault Generator can help diagnose an issue that

would not be evident at the physical layer. Implementing a Fault Generator within an FPGA is also more convenient than attempting to accomplish that on an external ATE.A self-test mode is also implemented to allow the system to calibrate itself before testing the UUT. The test can be accomplished with the instrumentation available in the FPGA-HST of Figure 3.

**High-Speed Failure Modes**

Table 2 lists some common failure modes, causes and effects of high speed I/O buses. These failure modes are mostly in the Physical (PHY) layer and more likely to occur with handling and assembly issues associated with system manufacturing or in the repair facility. Though not a complete list of all possible failure modes, they do represent most failure modes that are common to all types of high speed I/O buses.

Our FPGA-based instrumentation approach allows for testing higher level link and protocol stacks for a particular bus. In cases where the lower PHY level tests indicate no faults, the UUT may still suffer from failures that affect its ability to communicate or interface with other UUTs at the link- and protocol-layer levels. The FPGA can stimulate the bus with fault injected protocol (e.g. erroneous cyclic redundancy check (CRC) or error-correcting code (ECC) bits) to test the UUT's ability to correctly identify a failure mode that could occur in higher layers but would otherwise be imperceptible to PHY layer tests. Test instruments that deliver only "good" protocol stimuli to the UUT would not detect such failure modes. With fault injection at higher levels, such faults can be made detectable.

Our next step in the analysis is to develop specific tests to cover the failure modes in Table 2. The table does not contain any bus-specific failure modes. The failure modes listed are fairly common for all high speed I/O buses.

**Table 2 – Typical Failure Modes in High Speed I/O Buses**

| Structural Failure Mode | Causes | Effects |
|---|---|---|
| Single wire opens or resistive | Poor connector contact; Broken cable(s) | Poor differential signals, more bit errors, etc. |
| Twisted pair opens (or resistive) | Broken connector;<br>Broken cable(s) | High bit error rates, failed communication |
| Leakage between signals or between signal and shield | Moisture or liquid contamination; Cracked insulation; Zapped ESD structures | Poor differential signals, more bit errors, etc. |
| Shorts between signal pairs | Cracked insulation;<br>Trapped conductive material in connector | Signal degradation with involved wire pairs, increased bit errors |
| Leakage to Pwr/Gnd | ESD damage | Reduced differential, more bit errors |
| Clk failure | Crystal failed due to vibration; PLL not functional | Failed communication |
| Tx drive weaken | Circuit aging;<br>Overstressed (previous Tx shorting or power spiking) | Reduced differential, more bit errors |
| Rx sensitivity weaken, CDR failures | Circuit aging;<br>Overstressed (power spiking) | Reduced differential, more bit errors |
| Power transient, glitches | Power system instability; Power shorting;<br>Other parts of system failure and draining | Higher level of jitter, more bit errors |

As evidenced by Table 2, it is not easy to find the root causes since many failure modes lead to the same symptom/effect. So, additional tests are needed to be able to identify the root cause. Table 3 lists the diagnostic flow from failure symptoms to possible causes and provides a framework for approaching the test development.

**Table 3 - Difficult to Diagnose Symptoms**

| Failure Symptoms | Possible causes | Approaches |
|---|---|---|
| Intermittent, higher BER | Many possible causes. See effect column in previous table (Table 2) | Test flow to progressively test/verify structural integrity to circuit functionality |
| System failed to communicate at all | Many possible causes | Break down system into sub-system and test individually until bad subsystem can be identified |
| Individual link tested good, but still fails occasionally in system | Link-to-link interaction (signal coupling, crosstalk, EMI) | Progress from single link tests to multi-link test; test for system power transient/stability in use; eliminate possible EMI sources (like high power equipment nearby)<br>The "Fault Generator" is instrumental in finding the root cause of this failure. |

The first two items of Table 3 cover failure modes in individual link (a single differential pair). A system would likely consist of multiple links (e.g. PCIe, Ethernet, etc.). Link-to-link interaction (last item in Table 3) can also lead to failure modes that cannot be identified through individual link testing.

**The Tests for High-Speed I/O**

In order to test the failure modes identified in Table 2, we can perform the following test procedures:

- Self-Calibration Tests
    - These tests validate the (synthetic) instruments we will use. It will also verify that the cable is fault free by running shorts, opens and leakage tests.
- UUT DC Test
    - Ensures that UUT's SerDes is properly plugged in and there is no significant damage.
- UUT Eye Margin Test
    - Ensures that UUT's Tx is connected to the tester Tx test port. Then high-speed transmission is conducted with deterministic data in the pattern generator. The Checker determines whether there is a Match between the transmitted and received tests. At the same time the Eye Chart is also evaluated.
- Tx Jitter, Noise and BER(JNB) Test
    - Starting with the UUT Eye Margin test setup, we adjust (margin) the Eye in order to find passing unit interval (UI), as well as amplitude margins. (A UI is the minimum time interval between condition changes of a data transmission signal).
- UUT Rx Jitter Tolerance Test
    - Tests how the Rx reacts to changing jitter.

With traditional test methods, test engineers would have to select, procure and utilize general-purpose instruments that can apply the proper stimuli and make appropriate measurements with sufficient accuracy. With our approach, they would implement the design of SIs within the FPGA to accomplish the tests. However, to the extent that such SIs already exist for the FPGA, the test engineer could take advantage of instrument and test routine reuse. In the long run to the industry, reusing

test routines, SIs and test sets will reduce the overall test engineering effort and lead time substantially while also providing a better test at a fraction of the costs, with the added bonus of improved time-to-market.

**Summary and Conclusions**

Since an ATE is permanently configured at the time it was designed, it is not flexible to keep up with UUT technology improvements. Test instruments may be updated more frequently, but they are also years behind UUT technology advancements. Using FPGA-based Sis instead of, or in addition to the ATE test instrument, is a novel concept because FPGA technology is much closer in specifications to UUTs we are likely to encounter in today's production.

It is especially appropriate for high-speed I/O test applications. Traditional instrumentation technology suffers from the need to be general-purpose so that more buyers can absorb the cost of development. For a functional ATE to be able to test today's high-speed I/O it needs to be flexible and dynamic, so it can readily change with the need for ever-increasing speed and complexity. Because this is a monumental requirement, we have seen most ATE suppliers shy away from the functional board test (FBT) market. Instead, test engineers are tasked not only with test program development but also with designing and integrating the collection of instruments they need to create their own ATE. This often cuts into their test development times and often require lead times that can impact time-to-market.

In this paper, we propose that the SI in an FPGA is a better solution. Not only does it improve access to the UUT by being physically closer and reducing impedance and noise, but we believe this approach reduces test engineering effort both in the short term and in years to come. In the short term, it offers test engineers more capable instrumentation than those available from general-purpose designs. While complex SIs need to be designed, we believe they are more cost-effective than using that same time and effort in test program development. Even if this does not prove to be the case for every SI, the long-term benefits of reuse will reap a profit. As the test engineering community develops more SIs and more test sets, a library can be built that houses both SIs and tests. Test engineers will be able to search these libraries for tests and SIs they need for their applications. Whether sold or accessible from an open source, with this approach, test sets will be available that will be reusable, scalable and profitable for the most current and capable FPGAs available today and in the future.

As more test engineers use and reuse the library of tests and SIs, they will scrutinize them and thereby perform a peer review. If an error is found, that information can be fed back to the source of the reusable test from which prior and future users of that test can benefit. Others who have used those tests previously can be notified of mistakes found. Ultimately the veracity of reusable test routines will improve as newer revisions are created. The tests will become more accurate and comprehensive, resulting in better designs, manufacturing, test and greater customer satisfaction.

**Acknowledgements and Disclaimer**

Part of the research for this article was made with Government support under N68335-16-C-0434 and under N68335-18-C-0165 awarded by the Department of the Navy.

**References**

1. Ungar, Louis Y., "Design for Testability (DFT) to Overcome Functional Board Test Complexities in Manufacturing Test," Proc. IPC APEX, Feb. 2017.
2. IEEE, IEEE Std 716-1995 - IEEE Standard Test Language for All Systems - Common/Abbreviated Test Language for All Systems (C/ATLAS), 14 March 1995.
3. National Instruments, LabView 2018, http://www.ni.com/en-us/shop/labview/labview-details.html
4. IEEE, IEEE 1641-2010 - IEEE Standard for Signal and Test Definition, 17 Sep 2010.
5. IEEE, IEEE 1687-2014 - IEEE Standard for Access and Control of Instrumentation Embedded within a Semiconductor Device, 5 Dec 2014.
6. IEEE, IEEE 1671-2010 - IEEE Standard for Automatic Test Markup Language (ATML) for Exchanging Automatic Test Equipment and Test Information via XML, 20 Jan 2011.
7. IEEE, 1671.2-2012 - IEEE Standard for Automatic Test Markup Language (ATML) Instrument Description, 15 Feb. 2013.
8. Marvin Rozner Jr., Synthetic Test Systems, The Future of Test – Available Today, Aeroflex 2003. http://www.eetimes.com/document.asp?doc_id=1196003
9. C. T. Nadovich, Synthetic Instruments, Concepts and Applications, Elsevier Inc., 2005.

10. DeepekBehara, et al., Eye Diagram Basics: Reading and applying eye diagrams, EDN, December 2011.http://www.edn.com/design/test-and-measurement/4389368/Eye-Diagram-Basics-Reading-and-applying-eye-diagrams
11. Abhijit Athavale and Carl Christensen, High-Speed Serial I/O Made Simple: A Designers' Guide, with FPGA Applications, Edition 1.0, Xilinx, 2005.
12. Wang, Laung-Terng, et. al.. System on Chip Test Architectures: *Nanometer Design for Testability*. New York: Elsevier Inc, 2008.
13. Athavale, Abhijit and Christensen, Carl, High-Speed Serial I/O Made Simple: *A Designers' Guide with FPGA Applications*, Edition 1.0, Xilinx, 2005.
14. Ungar, Louis Y., "Testable and diagnosable commercial off the shelf (COTS) electronics," Proc. AUTOTESTCON, Sep. 2011
15. iLarslan, M. and Ungar, L., "Mitigating the impact of false alarms and no fault found events in military systems," IEEE Instrumentation & Measurement Magazine, Aug 2016, pp. 15-21.
16. Ungar, Louis Y. and Sudolsky, Michael D., "Tapping into boundary scan resources for vehicle health management," Proc. AUTOTESTCON, Sep. 2016.