Error Vector Magnitude (EVM) Testing: Improving Throughput and Flexibility Using FPGA Augmentation

Mark Roos – President & CEO, **Roos Instruments**
Devin Morris – Applications Engineer, **Roos Instruments**

ABSTRACT - Presented is an ATE test solution that employs an FPGA to supplement DSP requirements and data throughput for fast measurement of EVM. This approach seeks to enhance tester flexibility to support new and emerging test standards and devices. A general description of EVM is provided in addition to an overview of the test module architecture. The role of the FPGA and subsequent DSP architecture is provided to illustrate areas of high-speed processing demand and demonstrate strategies for reduced test time and enhanced data throughput. A case analysis of two different DSP implementations that compute differential EVM on a captured baseband Bluetooth signal are presented. A comparison of their performance illustrates non-ideal hardware and digital impairments both from a signal processing and deconstruction standpoint, as well as their subsequent EVM results for tester architecture considerations and DSP design edification.

INTRODUCTION

With the complexity of integrated radio designs and system on a chip (SoC) architectures becoming more prevalent, the demand for a strategy that enables efficient and reliable test of sub-system components and system-level verification has become necessary. Mitigating the added complication of embedded test points and extensive test time has becomes paramount to controlling cost in this environment. This has lead to the wide acceptance of Error Vector Magnitude as an important measurement in qualifying radio operation. There has been increased interest in implementing it in a production test environment because an EVM test offers the advantage of minimal I/O requirements, test signal stimuli that replicate "live operating conditions," and a standardized measurement that qualifies the digital and analog/RF operation of a device under test within a single test metric. However, this measurement requires processing of substantially large data sets and places considerable DSP requirements on the tester to address the highly intensive computational load of evaluating EVM. To offset these prerequisites, an ATE strategy that employs an EVM test module with an integrated FPGA co-processor is proposed. This approach provides a flexible platform to support new and emerging radio standards as well as providing a high performance DSP conduit between the data capture hardware and PC controller/interface to improve data throughput in computing EVM.

BACKGROUND

EVM is a measurement that directly links the digital and analog components of a radio by means of quantifying the signal's modulation accuracy in the symbol constellation domain. Groups of digital bits are represented as symbol points in this Cartesian coordinate system where the Inphase and Quadrature (IQ) analog signals serve as the x and y variables with a relationship of $\mathbf{2^n}$ number of symbols representing **n** number of bits, see Figure 1. During the modulation of the data, digital, analog/RF and random noise impairments manifest themselves in the symbol constellation as deviations from the ideal symbol points. By quantifying this deviation, it is possible to verify the operation of the entire modulation system within a DUT by analyzing its output modulated signal. Additional analysis of the deviation's behavior and overall characteristics in the constellation domain allows for identification of individual impairments such as:

- Amplifier non-linearity
- LO leakage
- Channel interference
- Quantization noise
- Quadrature offset
- IQ imbalance

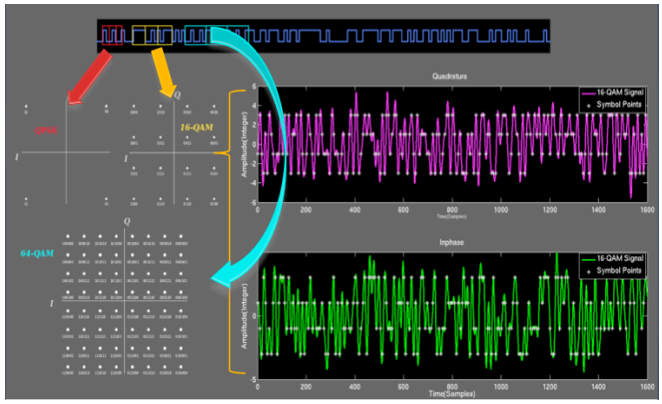

Figure 1: Translation of digital bits to analog signals in a modulation scheme: digital bits grouped according to modulation scheme (top); symbol constellation diagram of the corresponding bits (bottom-left); corresponding Inphase and Quadrature analog signals with included symbol points (white dots) for 16-QAM (bottom-right)

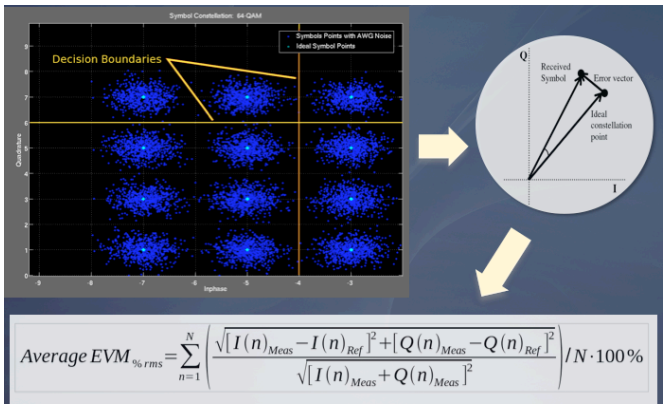

Figure 2: Calculating EVM: Constellation representation of recovered symbol data (top-left); Vector representation of EVM (top-right); Mathematical representation of EVM in terms of averaged values (bottom)

The computation of EVM itself is fairly straightforward: The recovered symbols measured from the DUT are subtracted from their ideal/reference symbols in the constellation domain. The magnitude of this difference vector is computed to produce the EVM of a symbol, which is typically averaged over several symbols to give an overall perspective of the DUT's system performance. In order to adequately characterize the DUT, the measurement is generally executed with a random bit pattern as the stimuli signal that spans the entire set of symbols in the constellation and exercises the DUT in a manner similar to how it would normally operate. The difficulty in making this measurement is the time and resource consuming nature of capturing and deconstructing the IQ signals from the DUT down to the symbol points needed for computing EVM. By inserting an FPGA into the demodulation chain, it is possible to mitigate the large data burden and subsequent DSP of the measured signal. This has the potential to reduce the processing time from tens or hundreds of milliseconds down to microseconds.

HARDWARE ARCHITECTURE

Figure 3 demonstrates the overall hardware setup and general signal flow in the testing of transmitted EVM using an FPGA. The data processed to compute EVM are captured from the output of the DUT with a complex detector which serves as a wide band down converter of both the I and Q paths. Once digitized, the demodulation of the data down to the necessary IQ symbol points is left to the FPGA. There are two distinct advantages with this approach: (1) can take advantage of the FPGA's high speed I/O to stream digital data directly to the signal processing stage, and (2) utilize the dedicated DSP resources to process the data in real-time. In order to facilitate this large reduction in test time, portions of the digital demodulation process were evaluated for implementation in an FPGA by whether:

- The demodulation component requires intensive processing
- It can be sufficiently reconstructed within the FPGA's limited DSP resources
- The ability to reconfigure or re-task the component/s provides adequate flexibility to support multiple applications.

Pulse-shaping filters, IQ compensation/correction, symbol timing recovery, and the EVM measurement meet one or several of these criteria, and are the focus of interest in terms of implementation.

As will be demonstrated in the subsequent case analysis, rigorous design control over the tester hardware and DSP techniques are key in preventing impairments created by the test equipment from skewing the EVM measurement. Though the stages in the digital demodulation are clearly defined, there are a multitude of DSP techniques and methodologies to execute these mechanisms. This point, coupled with the complex interplay of the hardware and DSP stages in the

tester create a difficult environment to indentify and characterize errors generated internally. Demonstrated in the following section is an example of this type of systemic error in the tester procedure.

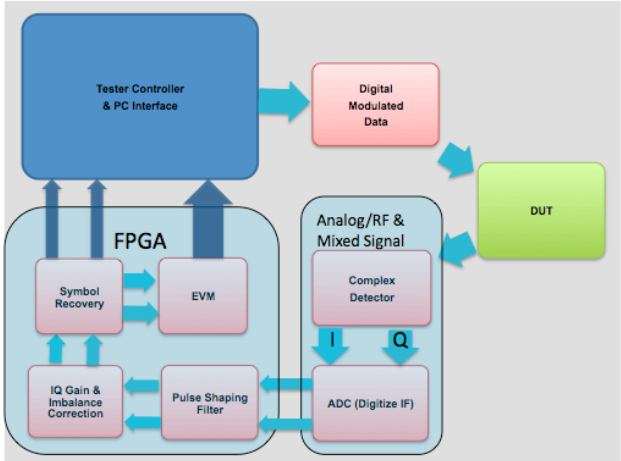

Figure 3: Overview of the architecture of the ATE solution for EVM measurement

CASE ANALYSIS

In the follow case study data an 8DPSK baseband Bluetooth transmitter DUT was digitized using a Teradyne tester and provided for signal analysis. The data was input to two different DSP techniques for signal demodulation and computation of differential EVM (DEVM) in accordance to Bluetooth specifications. The demodulation method used by the customer's test plan (Method B) was a MATLAB script that would accept a captured data stream from the tester, demodulate the data and compute DEVM. The methods in this routine were compared against a MATLAB/Simulink demodulation method and DEVM/EVM measurement developed at Roos Instruments (Method A) to be implemented in an FPGA. The focus was to identify and characterize impairments in the customer's DUT, differentiate them from impairments in the demodulation process, and quantify the effects in term of DEVM/EVM for both approaches. The comparison of the DSP techniques and analysis identify and discern test errors in both the hardware and software paths and provide insight into optimizing the demodulation process in the tester for more accurate measurement. As well, the step-by-step process taken to deconstruct the impairments and identify the sources of error in the measurement are presented to outline a debugging strategy for this type of testing and to provide merit to the procedure. As will be demonstrated, impairments introduced by the data capture process and artifacts created by the demodulation algorithms severely affected the resulting DEVM/EVM measurement.

The guidelines that both methods of digital demodulation and DEVM used were based on those outlined in the Bluetooth Specifications: Version 2.1. However there were some differences between them, most notably the partitioning of the data set for processing and the methodology for signal correction. Method B partitioned the data into blocks of 50 symbols for processing, setting correction values for IQ imbalance, frequency and phase offset using a constrained nonlinear multivariable function solver to minimize the resulting DEVM. This used a recursive algorithm to optimize the correction factors for each set of 50 symbols. Below is the code's method flow:

1) Input data from file
2) Apply Filter
   a) Square Root Raised Cosine
3) Synch Data Stream (similar to carrier recovery)
   a) Use preamble
4) Remove preamble from data
5) Partition data into packets of 50 symbols
6) Frequency Offset Correction
   a) Recursive algorithm optimizing for minimal EVM/DEVM
7) Phase Offset Correction
   a) Recursive algorithm optimizing for minimal EVM/DEVM
8) IQ Imbalance Compensation
   a) MATLAB Optimization Toolbox algorithm based on minimal EVM/DEVM
9) Compute and output EVM/DEVM

Method A processed the correction factors of IQ imbalance, and phase offset with manually adjustment by the user across the entire set of data before partitioning the recovered symbols into blocks of 50 for computing DEVM. This was done as a precursory debugging strategy in order to allow for user control in adjusting the correction factors while the data was being demodulated. By keeping the correction factors constant across the entire set of data, this prevented artifacts created by

the partitioning and compensation techniques from masking and or skewing behavior inherent in the capture system, demodulation process and/or the DUT impairments. Once the behavior and errors of the tester, DUT and demodulation algorithms could be identified, uncoupled from each other and then removed, the data could then benefit from phase, frequency and gain correction according to Bluetooth Specifications.

As stated before, the Simulink demodulation system used in Method A utilized components that translate directly to their FPGA counterparts. The demodulator was comprised of several arithmetic and communication blocks that together comprised a complete system for recovering the data, correcting for impairments, and computing DEVM/EVM in a similar fashion to Method B with the additional test metrics provided by a vector signal analyzer for debugging purposes. The preamble was removed from the data, and the data was processed as one large set of streaming data. The demodulation tool's correction/compensation arithmetic could be accessed during simulation runtime and served as a synthetic instrument of a vector signal analyzer. With this analysis environment it was possible to view, measure and manually correct for: Quadrature offset error, IQ imbalance, Gain adjustment, DC offset correction, Magnitude Error, and Phase Error. The symbol timing recovery algorithm used was optimized for D8PDK.. The Squaring timing method was used for symbol recovery by estimating the symbol timing phase offset for a given sequence and outputting only the symbol spaced values by applying the estimate of the symbol location in the data uniformly over the sequence of data. For the captured data, there are 16 data points/symbol yielding 2793 symbols for the entire set of 44688 data points analyzed(after preamble removal). For this demodulation system, the data was processed in blocks 32768 data points, i.e. the largest integer multiple of 16 that was less than or equal to 44688. This was done to process as many points as possible with the same phase and timing recovery constants to reduce transient phenomena in the recovered symbols introduced by the demodulation system. This recovery technique kept the symbol-timing phase constant for the entire observation interval, making it ideal for the observation of short duration transients, or random phase perturbations such as jitter in the data capture process.

Initially the data demonstrated considerable quadrature offset, that later was shown to be caused by the capture system rather than the DUT. This example however demonstrates the initial difficulty in uncoupling the tester impairments in the demodulation flow from the DUT. Following are the correction factors used on the initial recovered symbols before DEVM/EVM was calculated:

Quadrature offset error:	32.4^{o}
IQ imbalance (I/Q):	1/1.14
DC offset error (I/Q):	-0.08/-0.04
Gain Adjustment:	11.4

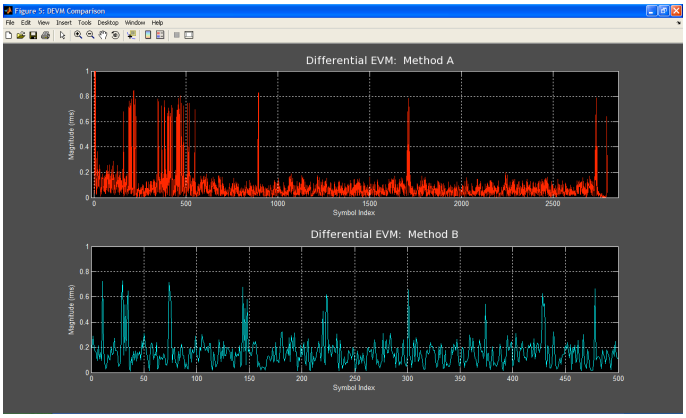

Figure 4: DEVM comparison between Method A (red) and Method B (cyan)

Once the symbols were recovered and error correction was completed, measurement of the remaining symbol error (DEVM/EVM) could be computed as seen in Figure 4, for both Method A and the Method B for comparison. The immediate discrepancy was the difference in the total number of symbols used for DEVM. Method B was limited to a total of 500 symbols for demodulation and DEVM/EVM, but the data was found to contain ~2793 symbols, excluding the initial preamble. This discrepancy was simply an internal processing preference to speed test time by limiting the data set to 10 groups of 50 symbol sets. Looking at the DEVM of Method A, it was significantly high initially and then reduces down to less than ~0.20rms with occasionally bursts of DEVM around 0.8rms. Method B had similar peaks of around 0.8rms periodic with the symbol sets of 50, and an average value around ~0.2rms. Due to the periodicity of the large spikes in Method

B's DEVM measurement, they were likely the product of the demodulation method/algorithm. Looking at the recovered symbol constellation points in Figure 5 further strengthens this claim.

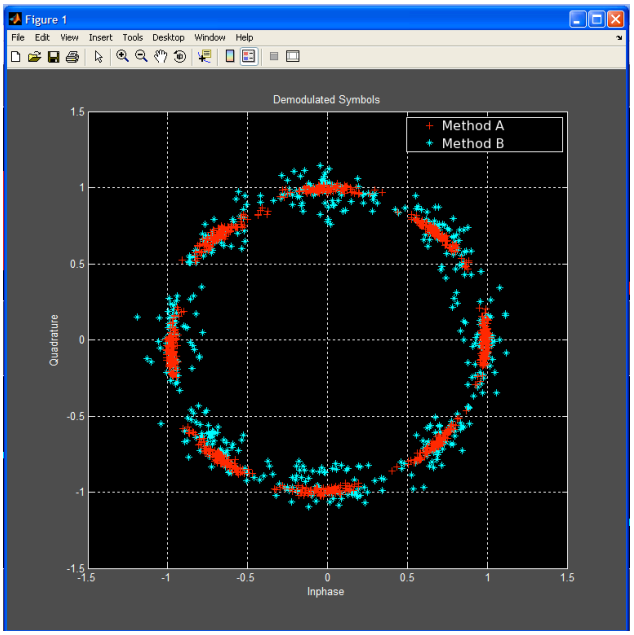

Figure 5: Demodulated symbols comparison between Method A (red) and Method B (cyan)

The clusters of symbol points from Method B have considerable more noise than the points demodulated using Method A. As a point of reference, in an ideal 8-PSK signal (no noise and/or impairments), the symbols points in the constellation would cluster around their ideal phase values: $\{0^o, 45^o, 90^o, 135^o, 180^o, 225^o, 270^o, 315^o\}$. It was difficult to explain the noisy behavior in Method B due to the compartmentalized/"black box" nature of the built-in MATLAB functions used to optimize phase/frequency offset and IQ imbalance. The error was most likely attributed to a combination of an insufficiently small data set to optimize across, coupled with a significant impairment(phase noise) in the data that renders the optimization to self-defeat. Suffice it to say that this approach of granting optimization algorithms free reign to seek a solution based on minimizing DEVM requires more oversight to prevent unpredictable results. Regardless of how the optimization technique arrived at the correction factor solution, this implementation yields a much more unstable and ultimately less optimal symbol recovery technique that was severely skewing any further conducted measurements downstream. However, looking at the recovered symbols from Method A, there was still a considerable amount of phase noise that was affecting the data, and the root cause of this has yet to be determined. In an effort to identify the cause of this, the following methodology was constructed.

With an impetus on isolating, and determining the nature of the phase behavior observed in the demodulated signal and large spikes in the DEVM measurement in Method A, a behavioral model was constructed using an identical demodulation system. Driven by an ideal Differential 8-PSK signal generator and user controlled impairment blocks to simulate IQ imbalance, quadrature offset, phase noise, carrier frequency drift, and signal noise. With this model, it was possible to emulate the behavior seen in the data from Method A, and extract measurements of the signal.

Using a phase noise generator and analyzing the symbol phase behavior and resulting DEVM/symbol, it was possible to recreate the noise behavior of the captured signal using 50.5dBc/Hz and SNR (Es/No) of 46dB. By analyzing each symbol "phase channel" it was possible to view the distribution of the symbol phase. In the case of phase noise, each symbol phase channel has essentially a Gaussian distribution about their respective ideal value. The phase range recorded in this experiment yielded a distribution about the phase channel's ideal value to be +/- 18^o with a $\sigma = 5.6^o$. This produced comparable DEVM behavior in terms of both average baseline DEVM(symbol regions with no spikes): Simulation – 0.0574 rms; Captured Data – 0.0509 rms, and with sufficient simulation runtime, random peaking of DEVM similar to that seen in the data of 0.8 rms. However, this failed to adequately explain the dense collection of spikes found in the beginning portion of the captured data DEVM in the symbol range of 0 - 1000.

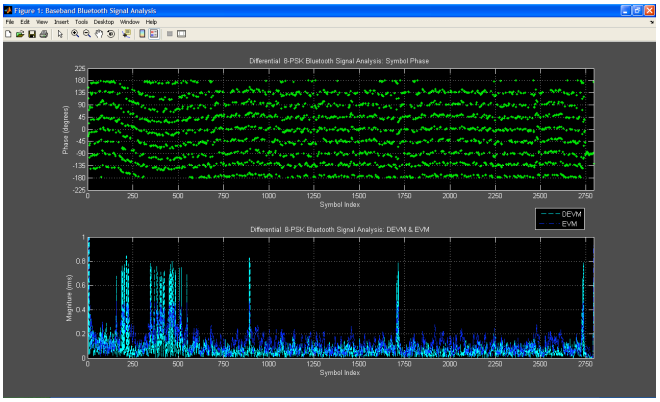

Figure 6: Captured symbol phase "sub channels" behavior and resulting DEVM/EVM vs. Symbol Index

With a mechanism to compare against in the behavioral model, the phase symbol sub channels were analyzed from the provided data. Comparing the phase relationship of the captured data, symbol for symbol with DEVM/EVM in Figure 6. It becomes quite apparent that a transient in the beginning portion of the data was causing the symbol phase in each channel to change, and affecting the EVM/DEVM. It's also worth noting in this specific case, there was a high correlation between EVM and DEVM ruling out a constant frequency/phase offset as a potential impairment culprit.

Employing the same strategy used to extract the phase behavior of a symbol "channel" in the simulation model, a subchannel of symbol phase points was extracted and used to produce a model of the phase behavior that was consistent across each subchannel. Since the accuracy of the interpolation was dependent on the number of points contained in each specific symbol "channel", and the data was pseudo random, the symbol phases occurring at -135^{o} were selected arbitrarily for their perceived high concentration across the entire signal range.

Several iterations of curve fitting were employed to seek an optimum solution to use as phase correction of the original data by subtraction of a normalized, mirror image of these phase function approximations from the phase of the original signal. In the following figures are graphs of the results of this methodology.

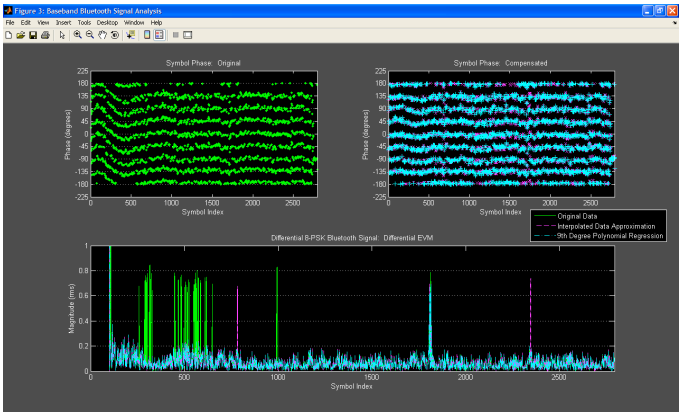

Figure 7: Comparison of Original Phase Symbols (top-left) & DEVM (bottom) vs. Phase Corrected Data Symbols (top-right) & DEVM (bottom): First Iteration – Interpolated function; 9th Degree Poly. Regression

In Figure 7, the first two function iterations are demonstrated: the raw interpolated data approximation, and a 9th degree polynomial regression best-fit line of the interpolated data. Comparing the phase behavior of the original data vs. the phase corrected using these two types of curves shows a considerable improvement in removing the transient. This was also reflected in the DEVM, with the initial peaks now significantly reduced. The peak around symbol index 1850 was still present in all cases, as well as new peaks created by the interpolated function correction around symbols 750 and 2350. These new peaks are most likely a byproduct of constructive phase noise error caused by the addition of the "noisy" phase approximation of the interpolated method. Note that the polynomial regression, which employs a smooth function approximation, eliminates these spurious values of DEVM.

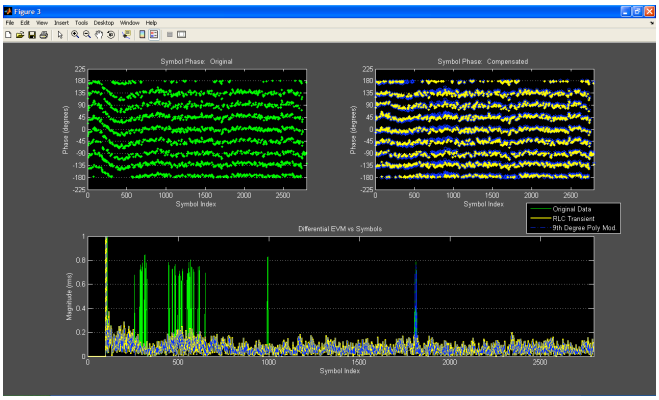

Figure 8: Comparison of Original Data Phase & DEVM vs. Phase Corrected Data Phase & DEVM: Second Iteration – Modified 9th Degree Poly. Regression; RLC Transient Response

In the second iteration of phase correction, a modified 9th degree polynomial regression and an RLC transient approximation are used with the results shown in Figure 8. Both these iterations were able to eliminate the spurious peaks in the DEVM measurement, with the RLC transient correction eliminating all of the spikes. The RLC correction process demonstrated a high likelihood that these anomalies were a product of the tester hardware, and most likely caused by a transient during start-up, test initialization, etc. Because the transient was severe enough initially, and the decay rate was roughly 25% of the entire data set, the overall effect in the constellation domain mimicked quadrature offset of what was previously reported of roughly 32°. The phase correction of the transient via the recovered symbols resulted in the mitigation of this measured quadrature offset, but a remaining phase noise of approximately 9% could now be linked to the DUT with a high degree of certainty. The RLC phase correction dramatically improved the peak DEVM and EVM by roughly a factor of 3x from 85%rms originally to 29%rms. It was interesting to note that the large peaks in the DEVM/EVM of the original data would have constituted a failed part by virtue of the Bluetooth standards on acceptable DEVM/EVM; Looking at the average value of DEVM/EVM on the entire set of data, they contribute very little to the overall average when corrected (difference of ~1-2%). Over smaller sets of data, the contribution would certainly be more substantial, but this overall difference indicates the heightened sensitivity of this measurement even with averaging of over a 1000 symbols.

CONCLUSION

As seen in the example demonstrated, signal constellation analysis and EVM provide important feedback on the overall operation of a DUT, or as in this case a self-check mechanism of the tester. It also highlights the importance of a close design synergy between the hardware and DSP in an EVM tester in order to provide reliable results. With the added benefit of inserting an FPGA into the test module, considerable improvements in test time can be achieved by processing the large data sets in real-time with dedicated DSP and passing the results to the ATE controller PC. The continued increase in the signal processing capabilities in the FPGA coupled with supporting DSP development platforms such as MATLAB have provided a springboard for quickly and easily implementing various baseband processing. With the ability to reconfigure the baseband signal processing while maintaining the tester's hardware front-end, this architecture provides a flexible EVM test platform that can accommodate new and emerging standards.