Fast linearity testing methods for different DAC architectures based on segmented non-parametric models

Author list

***Abstract* — The production testing of analog circuit parametric specifications is a significant contributor to the overall cost of building analog and mixed-signal products. Data converters (ADCs and DACs) in particular are critical components of integrated circuits used in control/actuation and sensing applications, and their production test times often dominate the overall system-on-chip (SoC) test time if left un-optimized. In this paper, we specifically focus on static linearity testing of DACs and propose architecture-aware test methods that are combined with best-in-class fast linearity test concepts in the literature to optimize test time without compromising quality.**

The proposed methods exploit the hypothesis that the number of device errors which contribute to linearity errors can be captured by a significantly fewer number of variables than the number of codes at which linearity needs to be tested. We introduce a new method (called Extrapolated Reconstruction (ER)) based on the segmented model introduced in uSMILE, which provides a slightly more time and memory efficient way to estimate the DAC INL and DNL compared to uSMILE. We also demonstrate that the segmented model techniques fail to accurately estimate the INLs and DNLs of interpolated DACs since they do not account for interpolation. We thus develop an interpolated segmented model and enhance both uSMILE and ER to obtain two new methods that provide correct estimations for interpolated DACs. A linearity test time reduction of 15x-20x was seen in actual silicon measurement results for a 12b DAC and >100x was seen in multiple simulation case studies for 16b DACs.

I. INTRODUCTION

The trends of increasing design complexity propelled by the increasing levels of integration to drive the system BOM (bill of materials) low, increasing quality needs from customers especially in automotive and other low dppm (defective parts per million) markets, new process nodes and defect models, etc. are pushing cost of test to the forefront of chip development cost.

Data convertors - Digital-to-Analog-Converters (DACs) and Analog-to-Digital-Convertors (ADCs) - are key interfaces between the physical analog world and the digital world, and are widely used in mixed-signal integrated circuits today. With demand for high performing data convertors with increasing resolutions, the time required for testing them increases exponentially and hence, the test cost. In this paper, we specifically focus on the problem of optimizing the cost of testing DACs.

Many parametric specifications of the DAC may need to be tested before shipping a part out to the customer. One category is static linearity specifications, which involve measuring the integral nonlinearity (INL), the differential nonlinearity (DNL), offset and gain error. The other category is measuring the spectral performance of the DAC, including the Signal-to-Noise Ratio (SNR), Total Harmonic Distortion (THD), Spurious free dynamic range (SNDR), etc. [1], [2]. There are also transient characteristics like settling time, glitch impulse area, etc. Our work focusses on efficient testing of static linearity specifications of DACs.

Static linearity testing of DACs frequently dominates the overall test time of SoCs, and this directly translates to high test costs. Accurate testing of DACs in a time and cost efficient way is a very challenging task. Conventional DAC static linearity testing is done by sweeping the input DAC code from 0 to the maximum code, and capturing the analog output voltages with a digitizer. Depending on the expected level of noise due to various sources, multiple hits per code are usually required to average out the noise to acceptably low levels [2]. The testing time is long and the test equipment is expensive. These problems are only worsened as the resolution of the DAC increases.

In the past, many researchers have proposed methods to reduce the cost of DAC linearity testing. The proposed method in [3] used stimulus error identification and removal (SEIR) [4] to obtain the ADC linearity first and estimate DAC INL/DNL with the ADC. However, the DAC INL/DNL estimation accuracy is limited and the test time is long. In [5], a circuit with a deterministic dynamic element matching (DDEM) ADC and a dithering DAC was developed to test the DAC. A 14-bit DAC can be tested by an ADC with just 6-bit linearity using this method, but the proposed ADC circuit has to be used and it also suffers from the long test time problem. In [6], Huang, et al presented a static loopback testing technique for an ADC/DAC pair. The effective test resolution was raised by scaling and offsetting the DAC output. The effective DAC resolution was raised by scaling down the output during ADC testing. Conversely, the effective ADC resolution was raised by scaling up the DAC output during DAC testing. This method is only applicable for a segmented current steering DAC and it still has a long test time. In [7], Ting, et al, tested the current-steering DAC by measuring the major transition current difference with a current-controlled oscillator and counter. This method is fast and low-cost but it

is highly architecture-dependent. Although many other BIST based methods have been proposed for DAC linearity testing to reduce cost [7]–[11], not many methods have been investigated to reduce the test time and cost by reducing the number of samples itself that need to be measured.

The counterpart to the DAC, the Analog-to-Digital Converter (ADC) faces many of the same challenges for linearity testing, and thus, it is instructive to investigate and possibly, port over some of the techniques which have been introduced to reduce the time and cost of static tests for ADCs to DACs. The ultrafast Segmented Model Identification of Linearity Errors (uSMILE) algorithm was first introduced for ADCs in [12]. The method significantly reduced the number of hits per code required to test an ADC by using a segmented non-parametric model for the INL. The USER-SMILE method [13], [14] then combined this method with SEIR to additionally relax the linearity requirement on the input ramp for ADC testing. This algorithm was used as the inspiration to develop the uSMILE-ROME method for DAC testing [15], which solved the dual challenges of reducing the number of samples to be taken as well as reducing the test cost by enabling use of low-linearity on-board/on-chip digitizers as opposed to high accuracy digital voltmeters. The ROME or "Removal of Measurement Error" method relaxes the linearity requirement of the digitizer which is used to measure the output voltages of the DAC. However, it requires the ability to add a constant voltage shift between the DAC and the ADC. Although implementing the voltage shift is not very hard, sometimes, this facility is unavailable. A high accuracy digitizer will have to be used to measure the output. For such cases, the segmented model can still be applied independently to reduce the number of measurements that need to be taken.

This paper will focus on reducing test time by reducing the number of samples to be measured. This will be done by reviewing and re-visiting the basic idea behind the uSMILE algorithm and developing an alternative albeit equivalent Extrapolated Reconstruction method which can be used for various DAC architectures. Additionally, it is shown that the uSMILE algorithm gives inaccurate results for architectures which involve interpolation. Hence, a new interpolated segmented model is developed for accurate linearity testing of these DAC architectures. Based on this new model, 2 new methods, namely, the ultrafast Interpolated Segmented Model Identification of Linearity Errors (uISMILE) method and the Extrapolated Reconstruction with Interpolation (ER+I) method are developed for interpolated DACs. Extensive simulations have been performed to demonstrate the different methods, and their applicability to different DAC architectures. Measurement results of a few representative 12 bit DACs show the effectiveness of the different methods in drastically reducing the static linearity test time for DACs over the conventional method.

The remainder of the paper is organized as follows. Section II reviews the conventional method and the uSMILE method and develops an alternative extrapolated reconstruction method for DACs. Section III presents some simulation results for the uSMILE and ER methods, and also illustrates the drawbacks and limitations of the previous methods for DAC architectures which have interpolation. Section IV develops new algorithms which are suitable for testing of such DACs and Section V presents simulation results for these new algorithms. Section VI demonstrates the effectiveness of the proposed methods with measurement results. Section VII and Section VIII give the discussion about on-chip measurement and conclusion respectively.

II. SEGMENTED MODELS FOR DAC LINEARITY

A. Conventional method and its drawbacks

The conventional method for testing the linearity of a DAC involves sweeping the input code from 0 to the maximum code, and then measuring the output voltages using a digitizer. The digitizer can be in the form of a Digital Voltmeter or a higher resolution ADC with significantly better specifications than the DAC under test. Whichever the case, multiple measurements per code are required to average out the noise. Once the output voltages have been measured, the output voltages are subtracted from either an end-point fit line or a best fit line, and divided by the average voltage difference between 2 consecutive codes to get the INLs at each code in units of LSBs. As the resolution grows, the number of input codes grows exponentially, and so does the number of measurements that need to be taken. There are 2^n input codes for an n-bit DAC. Let's say that h number of measurements need to be taken per code to average out noise. The total number of measurements would then be $h \times 2^n$. For a 16-bit DAC, with h=64, over 4 million measurements would need to be taken. At a sampling rate of, say 500KSPS, the data acquisition alone will take around 9 seconds! Even with multi-site testing, this will result in significant test time and cost per chip.

B. uSMILE : ultrafast Segmented Model Identification of Linearity Errors

The conventional method essentially treats the INL/DNL error at each code as unrelated to each other, and so, the number of variables to be estimated is equal to the number of DAC codes. This is highly inefficient. In reality, especially for high resolution DACs, the number of truly independent error sources due to non-idealities of analog components is much smaller than the number of codes. For example, take a 16-bit R-2R DAC. The number of resistor mismatches is just $2 \times 16 - 1 = 31$ which is dramatically less than $2^{16} = 65,536$. Although there will be many more error sources, it is true that a limited number of independent error terms are sufficient to capture the errors in the input output transfer curve of the DAC. In other words, all the INL/DNL errors are highly correlated and are deterministic functions of a much smaller number of independent errors.

This correlated nature of the INL/DNL DAC errors makes a strong case for a model based approach to DAC linearity testing. The uSMILE method models the DAC's INL curve with a segmented non-parametric model. The INL curve of the DAC is broken into many MSB segments according to the MSB (Most Significant Bits) value of the DAC input code. Take a 16-bit DAC for example. If 6 bits are used as the MSB, then the INL curve is divided into 64 different segments. Each of these segments has an error term associated with it, say $e_M(C_{MSB})$, where C_{MSB} ranges from 0 to 63. Each of these segments in turn can be further divided into smaller segments. Say the next 5 bits are used as ISB (Intermediate Significant Bits), then each MSB segment gets divided into 32 ISB segments, each of which has an error term associated with it, say $e_I(C_{ISB})$. If we stop the segmentation here, the variations within each ISB segment away from the ISB average values are captured by the 32 LSB errors (5 LSB bits). The error term associated with each LSB segment is denoted as $e_L(C_{LSB})$. The final INL value for code C will be

$$INL(C) = e_M(C_{MSB}) + e_I(C_{ISB}) + e_L(C_{LSB}) \quad (1)$$

Most DAC architectures are inherently segmented in this fashion, like binary weighted, R-2R, current steering, mDAC etc., and so, this segmented non-parametric model can be applied to the INL curve. This segmented model of the DAC's INL enables us to estimate the INLs with significantly fewer output voltage measurements because it drastically reduces the number of variables to be estimated. Let's say that we have an n-bit DAC, and we sweep the DAC input code from 0 to max code, while taking 1 measurement per code. We calculate the preliminary INLs at each code, which will obviously have a significant amount of noise. Let's say we do an nMSB-nISB-nLSB segmentation of the DAC. We can write equation (1) at every code, giving us $M = 2^n$ equations. The number of unknowns $K = (2^{nMSB} + 2^{nISB} + 2^{nLSB})$. Since this is an overdetermined system with more number of equations than unknowns, the method of least squares can be used to compute the unknowns. Let's define e_M as a column vector:

$$e_M = \begin{bmatrix} e_M(0) \\ e_M(1) \\ \vdots \\ e_M(2^{nMSB}-1) \end{bmatrix} \quad (2)$$

e_I and e_L are defined in a similar way. We can then write $e_M(C_{MSB})$ as:

$$e_M(C_{MSB}) = [0 \quad 0 \quad ... \quad 1 \quad ... \quad 0]e_M \quad (3)$$

with the 1 being placed at the $C_{MSB}+1$ location. The INL at code C can thus be written as:

$$INL(C) = [0...1...0...1...0...1...0] \begin{bmatrix} e_M \\ e_I \\ e_L \end{bmatrix} \quad (4)$$

with locations of the 1s depending on C_{MSB}, C_{ISB} and C_{LSB}. We can then combine all the equations at every code in matrix form as:

$$P = H \times e + noise \quad (5)$$

where P is the preliminary full code INL column matrix of length M, H is an $M \times K$ coefficient matrix which has three +1s in each row, the placement of which depend on the DAC code, and $e = [e_M \quad e_I \quad e_L]^T$ is a row matrix of length K. The unknown vector e can be estimated using least squares as:

$$\hat{e} \approx H_{inv} P \quad (6)$$

where $H_{inv} = (H^T H)^{-1} H^T$.

Once the unknowns are estimated, the full code noise-free INL vector F can be reconstructed as

$$F = H \times \hat{e} \quad (7)$$

The method of least squares naturally averages out the noise. The average number of measurements per unknown $= M / K$. We will call this the time saving factor (tsf). This also gives us the equivalent number of hits per code. For a 16-bit DAC with an 8-4-4 segmentation, $M = 2^{16} = 65535$ and $K = 2^8 + 2^4 + 2^4 = 288$. Hence, the equivalent hits per code $= 65535 / 288 \approx 228$ i.e. the estimated INLs using uSMILE with just 1 measurement per code should be as accurate as if we had taken 228 measurements per code and calculated INL using the conventional method. The uSMILE algorithm essentially takes a noisy INL estimation (P) as an input and gives a noise-free estimation of the INL (F) as an output. In this sense, it acts like a noise filter, based on the segmented model. Thus, uSMILE enables us to estimate the INL/DNL of the DAC at each code with a much-reduced number of samples. The time and space complexity of uSMILE is discussed in Section VII.

Note that this segmented model is not valid for string or thermometer-coded type architectures. For example, if you have a segmented 15-bit DAC implemented as a 7-bit thermometer coded resistor DAC and an 8-bit R-2R DAC, then the segmentation of the INL curve must be carefully chosen such that the MSB bits are greater than or equal to 7, since the thermometer coded part does not have a segmented architecture. For example, a 7-4-4 segmentation of the INL curve is valid for this DAC, and so is an 8-3-4 segmentation, but a 6-5-4 or 5-5-5 segmentation is not valid. As will be seen in the next section, the segmented model also cannot be applied directly to DACs which have interpolation.

C. *Extrapolated Reconstruction (ER) method*

The basic idea behind uSMILE is that a limited number of non-idealities determine the deviation from the ideal output

voltages of a DAC. If we can determine these accurately, then the INL at each code can be determined. When the architecture of the DAC is segmented by MSBs, ISBs and LSBs, we say that the INL at code C can be written as the error due to the MSB DAC plus the error due to the ISB DAC plus the error due to the LSB DAC. In other words, $INL(C) = e_M(C_{MSB}) + e_I(C_{ISB}) + e_L(C_{LSB})$. This is derived from the fact that the output voltage at code C is equal to the output voltage of the MSB DAC plus the output voltage of the ISB DAC plus the output voltage of the LSB DAC, or in other words,

$$V(C) = V_M(C_{MSB}) + V_I(C_{ISB}) + V_L(C_{LSB}) \quad (8)$$

This means that if we can estimate $V_M(C_{MSB})$, $V_I(C_{ISB})$ and $V_L(C_{LSB})$ accurately, then we can extrapolate and re-construct the output voltages for all codes! For both ADCs and DACs, the non-linearities are defined per code. Unlike in ADCs, where we do not know the output code that the ADC will produce, DAC linearity testing offers a unique opportunity – we can actually obtain the measurements for specific output codes. As we will see, we can measure the output of the DAC accurately at specific limited number of codes, and derive $V_M(C_{MSB})$, $V_I(C_{ISB})$ and $V_L(C_{LSB})$ from these measurements and thus reconstruct the output for all codes.

Let's say we do an nMSB-nISB-nLSB segmentation of the DAC. We define the output voltage at code k as

$$v(k_{MSB}, k_{ISB}, k_{LSB}) = V_M(k_{MSB}) + V_I(k_{ISB}) + V_L(k_{LSB}) = V(k) \quad (9)$$

where $k_{MSB}, k_{ISB},$ and k_{LSB} are the MSB, ISB and LSB codes of code k .

First, we measure the outputs at all the "MSB points", which are basically all codes k for which the ISB and LSB codes are 0. Let's call these measurements $v_M(j) = v(j,0,0)$ where j varies from 0 to $2^{nMSB} - 1$. Next, we choose any one MSB segment (all codes for which $k_{MSB} = m$, say), and then measure the outputs at all the "ISB points" in this segment (all codes for which $k_{MSB} = m$ and $k_{LSB} = 0$). Let's call these measurements $v_I(j) = v(m, j, 0)$ where j varies from 0 to $2^{nISB} - 1$. Finally, we choose any ISB segment in this MSB segment (all codes for which $k_{MSB} = m$ and $k_{ISB} = i$, say), and measure outputs at all these "LSB points". Let's call these measurements $v_L(j) = v(m, i, j)$ where j varies from 0 to $2^{nLSB} - 1$. We can now write the voltage at any code C as:

$$\begin{aligned}
\hat{V}(C) &= V_M(C_{MSB}) + V_I(C_{ISB}) + V_L(C_{LSB}) \\
&= [V_M(C_{MSB}) + V_I(0) + V_L(0)] \\
&\quad + [V_M(m) + V_I(C_{ISB}) + V_L(0)] - [V_M(m) + V_I(0) + V_L(0)] \\
&\quad + [V_M(m) + V_I(i) + V_L(C_{LSB})] - [V_M(m) + V_I(i) + V_L(0)] \\
&= v(C_{MSB}, 0, 0) \\
&\quad + v(m, C_{ISB}, 0) - v(m, 0, 0) \\
&\quad + v(m, i, C_{LSB}) - v(m, i, 0) \\
&= v_M(C_{MSB}) + v_I(C_{ISB}) - v_I(0) + v_L(C_{LSB}) - v_L(0) \quad (10)
\end{aligned}$$

Thus, we can "reconstruct" the output voltage at all codes by measuring the output voltages at just a few codes. We can then calculate the end-point fit/best fit INLs and DNLs for all the codes just like we would have done in the conventional method.

Let's take a 12-bit DAC as an example, with a two level 7-5 segmentation to make things easier. Every 32 codes form one LSB segment. First, we measure the outputs at all the MSB points i.e. at codes 0, 32, 64, … , 2048, 2080, … , and 4064 with multiple hits per code to average out the noise. Then we measure the outputs for one LSB segment. If we choose this at mid-code, that means we measure at codes 2048, 2049, … , 2080. From just these measurements, we can extrapolate and reconstruct the outputs for all the DAC codes.

For an n-bit DAC, if the conventional method requires measuring the output voltages at $M = 2^n$ number of codes with, say h hits per code to average out the noise, then the proposed Extrapolated Reconstruction (ER) method would require measurement at $K = (2^{nMSB} + 2^{nISB} + 2^{nLSB})$ number of codes with the same h hits per code. Hence, the time saving factor $tsf = M / K$. For a 16 bit DAC with 8-4-4 segmentation, this value is nearly equal to 228. Notice that the tsf for ER is exactly the same as the tsf for uSMILE. In uSMILE, we measure the output voltage at all the codes, with 1 measurement per code. In ER, we measure the output at significantly fewer codes, but with multiple measurements per code. To give equal noise averaging capabilities, the total number of measurements for both methods will be the same. Although both methods are equivalent with respect to the accuracy of estimation, advantages and disadvantages of each will be detailed in Section VII.

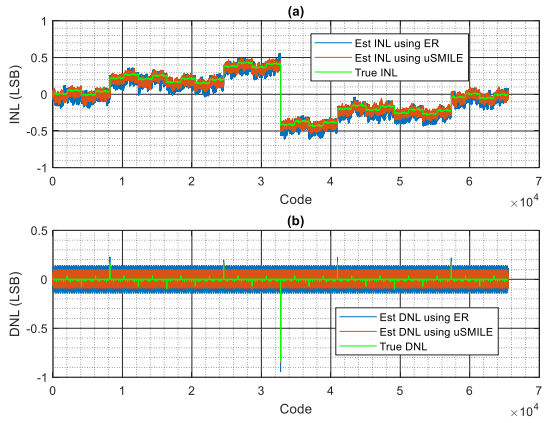

Fig 1. Simulation results of a 16b R-2R DAC
(a) INL estimations using uSMILE and ER
(b) DNL estimations

III. Simulation Results Part-1

We will now show the effectiveness of uSMILE and ER via simulations. First, a 16-bit R-2R DAC was modeled in Matlab with resistor mismatches. 0.5LSB of additive noise was added to the output. For simplicity, the DAC was segmented as 8-8 for uSMILE and ER. For ER, the output voltages were measured at $2^8+2^8=512$ codes, with 256 measurements per code. For uSMILE, measurements were taken at all 2^{16} codes, with 2 measurements per code. This makes the equivalent number of hits per code equal to 256, because $tsf=2^{16}/(2^8+2^8)=128$. The true and estimated INL curves are shown in figure 1(a), along with the INL estimation errors. Similarly, the DNL curves are shown in figure 1(b). We can clearly see that there is very good correlation between the true and estimated INLs and DNLs.

Next, a 16-bit segmented hybrid DAC implemented as an 8-bit thermometer coded resistor DAC and an 8-bit R-2R DAC, is modeled, with resistor mismatches, with the additive noise at 0.5 LSB level. With the same parameters as for the

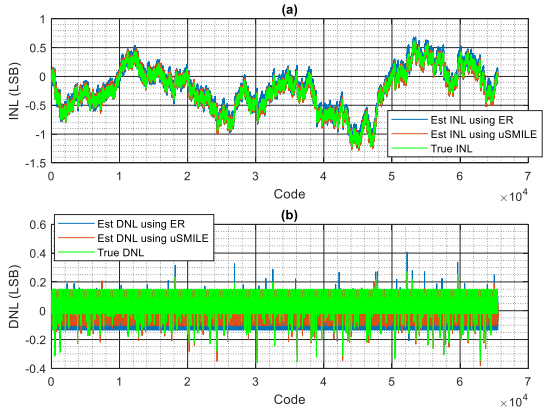

Fig 2. Simulation results of a 16b Segmented Hybrid DAC
(a) INL estimations using uSMILE and ER
(b) DNL estimations

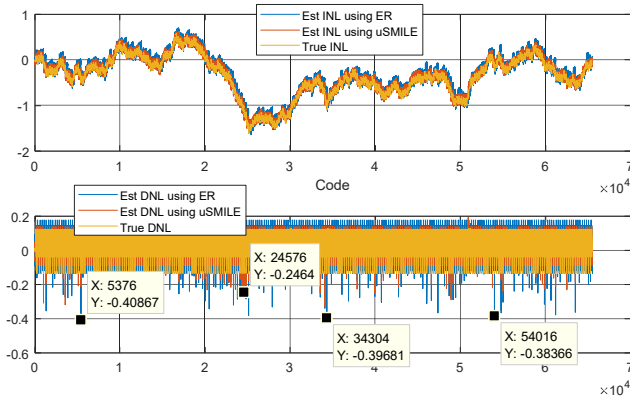

Fig 3. Simulation results of a 16b interpolated DAC
(a) INL estimations using uSMILE and ER
(b) DNL estimations

R-2R DAC, the INLs are estimated with the 2 methods. The true and estimated INL curves are shown in figure 2, along with the DNL estimation errors. Once again, the correlation is very good between the true and estimated INLs and DNLs. These simulation results validate that uSMILE and ER can accurately estimate the INL/DNL with a significantly reduced number of measurements.

To check their effectiveness for another DAC architecture, a 16b interpolated DAC is generated, with the 8-bit LSB R-2R DAC interpolating between the output voltages of the 8-bit MSB string DAC. The INL and DNL curves are shown in figure 3. In this case, it is clearly visible that the DNL estimations are inaccurate. We see that there are spikes in the DNL estimations at the MSB points. The reason for this will be explained in the next section. These figures indicate that the segmented model is not suitable for DACs which have interpolation.

IV. Interpolated Segmented Models for DAC Linearity

The simulation results in the previous section show that the uSMILE and ER methods give inaccurate estimations for the INLs and DNLs of DACs which have interpolation. This is to be expected because the segmented model does not account for interpolation. We see especially bad estimations at the MSB points, with large spikes in the DNL estimations at these codes. The reason for this is as follows. In the segmented model, we assume that the output voltage due to the ISB+LSB DAC gets added to the output voltage of the MSB DAC. But since the architecture of the interpolated DAC is such that $V(C_{MSB})$ and $V(C_{MSB}+1)$ effectively become the Vlow and Vhigh references for the subsequent ISB+LSB DAC, all the voltages get interpolated or scaled between these two voltages. Hence, in the segmented model estimations, there will be sudden jumps up/down in voltage when we go from one MSB segment to the other. Figure 4 gives a better visual representation of what is happening.

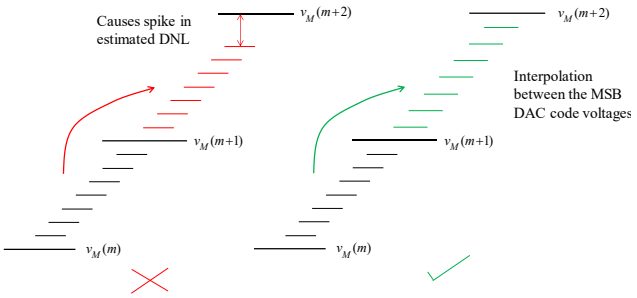

Fig 4. Segmented model estimation vs actual output voltages for a segmented DAC with interpolation

Given this deficiency of the segmented model, we need to come up with a modified version which accounts for interpolation.

A. uISMILE: ultrafast Interpolated Segmented Model Identification of Linearity Errors

We will now develop a model for the INL curve which is segmented and accounts for interpolation. In order to simplify the derivation of the equations, we will consider a 2 level nMSB-nLSB segmentation, assuming the LSB DAC interpolates between the MSB DAC output voltages. This LSB DAC can always be further segmented as required. Figure 5 shows a visual representation of how the output voltage V_{out} is being generated for some code C. Let's say that V_x is the output voltage of the LSB DAC for code C_{LSB} if the reference voltages were 0 and 1V. This voltage can be written as:

$$V_x(C_{LSB}) = \frac{C_{LSB}}{2^{nLSB}} + e_L(C_{LSB}) \tag{11}$$

where $e_L(C_{LSB})$ is the deviation from the ideal output voltage, or in other words, it is the INL in volts if the reference voltage of the LSB DAC were 1V. Similarly, the output voltage of the MSB DAC with reference voltage V_{ref} can be written as:

$$V_M(C_{MSB}) = C_{MSB} \times \left(\frac{V_{ref}}{2^{nMSB}}\right) + e_M(C_{MSB}) \tag{12}$$

Then, with the voltages $V_M(C_{MSB})$ and $V_M(C_{MSB}+1)$ from the MSB DAC being used as references for the LSB DAC, the output voltage V_{out} can be written as:

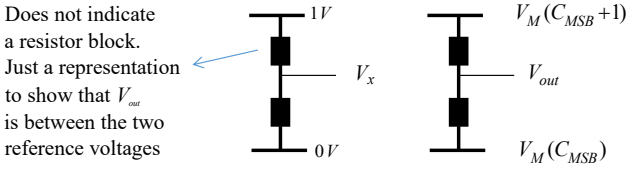

Fig 5. Visual representation of the LSB DAC and how the output voltage is generated in an interpolated DAC

$$V_{out} = V_M(C_{MSB}) + [V_M(C_{MSB}+1) - V_M(C_{MSB})] \times V_x \tag{13}$$

where $V_l = V_M(C_{MSB})$ and $V_h = V_M(C_{MSB}+1)$. V_{out} can also be written in terms of the INL of the whole segmented DAC as:

$$V_{out} = C_{MSB}\left(\frac{V_{ref}}{2^{nMSB}}\right) + C_{LSB}\left(\frac{V_{ref}}{2^{n}}\right) + INL(C) \tag{14}$$

Substituting equations 11, 12 and 14 in equation 13, we get:

$$\begin{aligned} &C_{MSB}\left(\frac{V_{ref}}{2^{nMSB}}\right) + C_{LSB}\left(\frac{V_{ref}}{2^{n}}\right) + INL(C) = \\ &C_{MSB} \times \left(\frac{V_{ref}}{2^{nMSB}}\right) + e_M(C_{MSB}) \\ &+ \begin{bmatrix} (C_{MSB}+1) \times \left(\frac{V_{ref}}{2^{nMSB}}\right) + e_M(C_{MSB}+1) \\ -C_{MSB} \times \left(\frac{V_{ref}}{2^{nMSB}}\right) - e_M(C_{MSB}) \end{bmatrix} \times \left(\frac{C_{LSB}}{2^{nLSB}} + e_L(C_{LSB})\right) \end{aligned} \tag{15}$$

After rearranging and cancellation of terms, we finally get:

$$\begin{aligned} INL(C) = {} & e_M(C_{MSB}) + \left(\frac{V_{ref}}{2^{nMSB}}\right) e_L(C_{LSB}) \\ & + [e_M(C_{MSB}+1) - e_M(C_{MSB})] \times \frac{C_{LSB}}{2^{nLSB}} \\ & + [e_M(C_{MSB}+1) - e_M(C_{MSB})] \times e_L(C_{LSB}) \end{aligned} \tag{16}$$

Since this INL is in units of volts, we can divide the whole equation by $V_{ref}/2^n$ where $n = nMSB + nLSB$, to get the INL in units of LSBs. Next, if we define $e_M'(C_{MSB}) = e_M(C_{MSB}) / (V_{ref}/2^n)$ and $e_L'(C_{LSB}) = e_L(C_{LSB}) \times 2^{nLSB}$, then we can finally write the model for INL as:

$$\begin{aligned} INL(C) \text{ in LSBs} = {} & e_M'(C_{MSB}) + e_L'(C_{LSB}) \\ & + [e_M'(C_{MSB}+1) - e_M'(C_{MSB})] \times \frac{C_{LSB}}{2^{nLSB}} \\ & + err \end{aligned} \tag{17}$$

where $e_M'(C_{MSB})$ and $e_L'(C_{LSB})$ are representative of the nonlinearities due to the MSB DAC and LSB DAC respectively, in units of DAC LSBs. The error term $err = [e_M'(C_{MSB}+1) - e_M'(C_{MSB})] \times e_L'(C_{LSB}) / 2^{nLSB}$ can be ignored since it is a multiplication of two nonlinearities and will be small. The coefficient of $e_M'(C_{MSB})$ is $\left(1 - C_{LSB}/2^{nLSB}\right)$ whereas the coefficient of $e_M'(C_{MSB}+1)$ is $\left(C_{LSB}/2^{nLSB}\right)$. This means that unlike in normal segmented DACs, where the INLs for codes in an MSB segment were dependent on one MSB code nonlinearity, the INLs for codes in an MSB segment for interpolated DACs are dependent on the weighted average of the 2 adjacent MSB code nonlinearities, with the weights depending on how close/far the code is from the MSB DAC codes.

Now that we finally have a segmented non-parametric model of the INL for interpolated DACs, we can follow exactly the same procedure as in uSMILE – we can calculate the preliminary INLs P and then write equation 17 for each DAC code, get the coefficient matrix H, put the equations in matrix form, and estimate the vector $e' = [e_M' \quad e_L']^T$ using least squares. The final estimated INLs F can be calculated as $H_{inv} \times e'$. It should be noted that although we used the ideal value of $1LSB = (V_{ref} / 2^n)$ for going from equation 16 to equation 17, the units of the estimated INLs will be whatever units the preliminary INLs are in (actual LSBs). The scaling factor will just be absorbed into the unknowns vector e'.

B. Extrapolated Reconstruction with Interpolation (ER+I)

Even for segmented DACs with interpolation, it is possible to extrapolate and reconstruct the output voltages for all codes by measuring the voltages at the same specific codes mentioned in Section II.C. We will just have to tweak the equation that we use for reconstructing the output voltages. We will again consider a two level nMSB-nLSB segmentation for simplicity. We first measure the output voltages at the MSB points, and call them $v_M(j) = v(j,0)$ where j varies from 0 to $2^{nMSB} - 1$. We then measure all the voltages in one LSB segment: $v_L(j) = v(m, j)$ where j varies from 0 to $2^{nLSB} - 1$. Now, to reconstruct the output at any code, we can simply re-use equation 13. But for that, we require $V_x(j)$ which are the output voltages of the LSB DAC when the reference voltages are 0 and 1V. Following logic similar to what is described in Section II C, but additionally accounting for interpolation, we can get $V_x(j)$ by subtracting $v_M(m)$ from $v_L(j)$ and dividing by $v_M(m+1) - v_M(m)$. Hence, all the output voltages can be reconstructed using the following equation:

$$\hat{V}(C) = v_M(C_{MSB}) + \left(v_M(C_{MSB}+1) - v_M(C_{MSB})\right) \times \left(\frac{v_L(C_{LSB}) - v_M(m)}{v_M(m+1) - v_M(m)} \right) \quad (18)$$

All the discussions about the time saving factor for the ER method are valid for the ER+I method.

V. Simulation Results Part-2

We will verify that the uISMILE and ER+I methods give accurate INL and DNL estimations for interpolated DACs. uISMILE and ER+I were used to estimate the INLs of the same interpolated DAC that was generated in Section III, with the additive noise set to 0.5LSBs. As before, for ER and ER+I, 256 measurements were taken per code, whereas for uSMILE and uISMILE, 2 measurements per code were taken. Figure 6 shows the estimated INL plots using all 4 methods, compared with the true INL, and figure 7 shows the DNL plots. The estimation errors are summarized in Table I. We

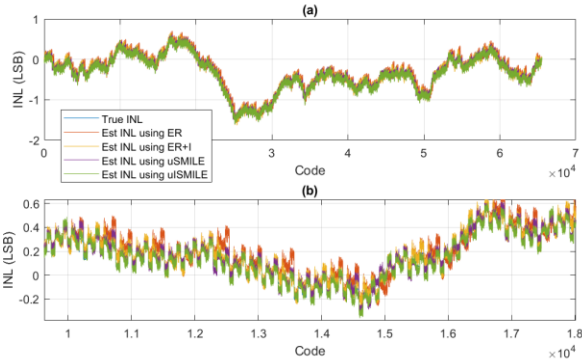

Fig 6. INL simulation results of a 16b interpolated DAC
(a) INL estimations using the different methods
(b) Zoomed in view of the INL plots

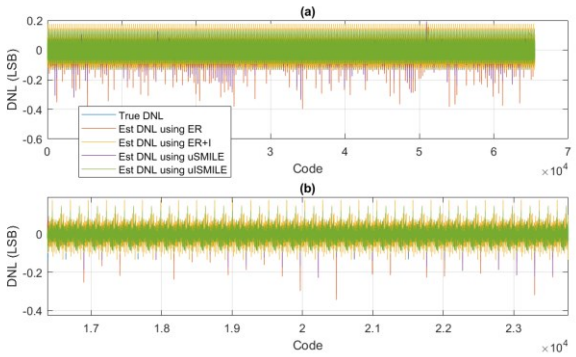

Fig 7. DNL simulation results of a 16b interpolated DAC
(a) DNL estimations of the 16b DAC using the different methods
(b) Zoomed in view of the DNL plots

can clearly see the jumps at the MSB codes in the DNLs estimated by the algorithms which do not account for interpolation i.e. uSMILE and ER, whereas uISMILE and ER+I give good estimations for both the INL and DNL. To verify the robustness of the ER+I and uISMILE methods, 100 16b DACs were randomly generated. Figure 8 shows a scatter plot with the true absolute maximum INLs/DNLs on the X-axis and the estimated absolute INLs/DNLs on the Y-axis. There is excellent correlation between the true and estimated INLs and DNLs for both the methods.

Table I Simulation results of a 16b interpolated DAC
(a) INL estimation errors in LSBs

	Max(abs(INL_est-INL_true))	(INLmax_est-INLmax_true)	(INLmin_est-INLmin_true)
ER	0.36	0.131	-0.027
ER+I	0.19	0.078	-0.026
uSMILE	0.25	0.039	0.030
uISMILE	0.12	0.034	-0.006

(b) DNL estimation errors in LSBs

	Max(abs(DNL_est-DNL_true))	(DNLmax_est-DNLmax_true)	(DNLmin_est-DNLmin_true)
ER	0.29	0.051	-0.27
ER+I	0.14	0.052	-0.002
uSMILE	0.33	0.068	-0.19
uISMILE	0.09	0.014	0.038

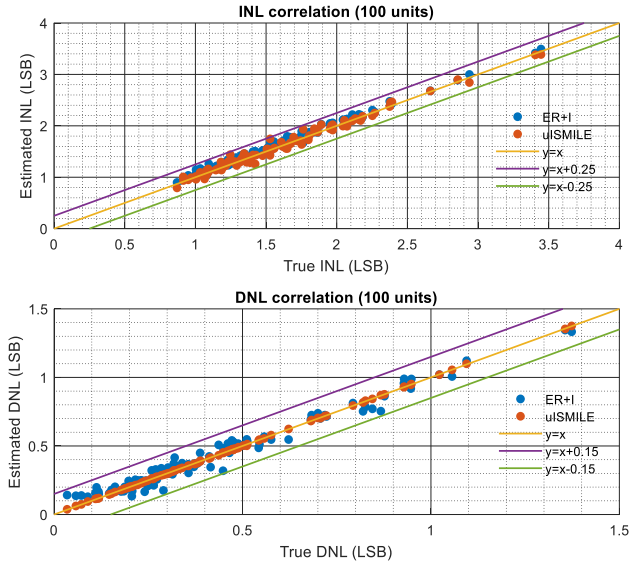

Fig 8. Simulation results – Robustness test for 16b interpolated DACs
(a) INL correlation using uISMILE and ER+I
(b) DNL correlation using uISMILE and ER+I

VI. Silicon Measurement Results

The uSMILE algorithm has been shown to work in simulations in the previous sections, and in practice for ADCs in previous literature. The focus here will be to show the effectiveness of the uISMILE algorithm and the Extrapolated Reconstruction with Interpolation (ER+I) method with measurement results on a production mixed-signal IC. This IC has a 12b DAC and a higher resolution ADC available on-chip. The 12b DAC has an m-n segmentation, with interpolation between the output voltages of an m-bit MSB DAC. The ADC was used to measure the output of the DAC.

Since the linearity of this DAC is specified for a sub-range of codes, the end-point fit INLs/DNLs were calculated and compared in this range. For uSMILE/uISMILE, the DAC input code was swept from min code to max code. From a simple calculation like the one presented in previous sections,

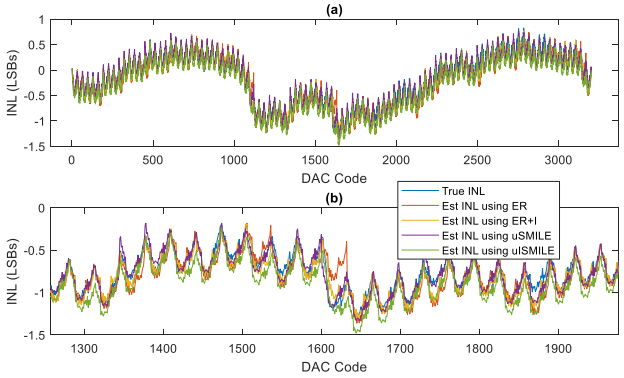

Fig 9. INL Measurement results of a 12b interpolated DAC
(a) INL estimations using the different methods
(b) Zoomed in view of the INL plots

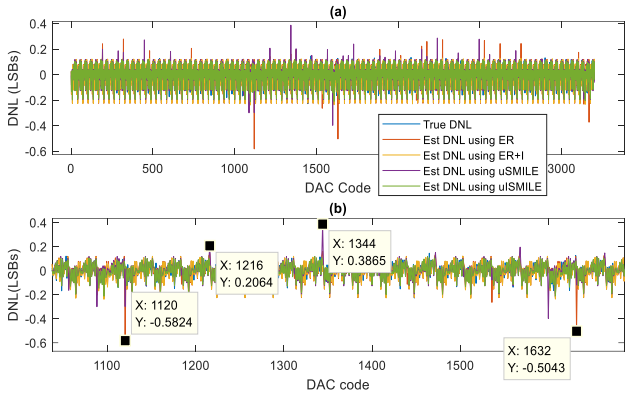

Fig 10. DNL Measurement results of the 12b interpolated DAC
(a) DNL estimations using the different methods
(b) Zoomed in view of the DNL plots

it is found that the $tsf \approx 20$ i.e. if one were to take one measurement per code, the equivalent hits per code for uSMILE/uISMILE would be ~20. Hence, 12 measurements were taken per code to make the effective number of hits per code 240. For the ER/ER+I methods, the outputs for only the MSB codes, and one LSB segment were measured, with 240 hits per code to make it comparable to uISMILE. The LSB segment was chosen at mid code (2048). For the conventional INL, the DAC input code was swept from min code to max code with 240 measurements taken per code. The data acquisition time for all the fast linearity methods is 20x less than the conventional method. To ensure that the impact of noise is minimal, the conventionally measured INL was averaged across 10 runs to get "true" INL with an effective hits per code of 2400.

The INL and DNL per code were estimated using the 4 different algorithms (ER, ER+I, uSMILE, UISMILE). The complete true and estimated DAC INLs per code are shown in Fig 9 (a). Visually, all the estimation methods seem to track the INL well and give pretty accurate results. Fig 9 (b) shows a zoomed in view of the plots over specific codes, where the INL estimation using ER (without interpolation) deviates slightly from the true INL in one LSB segment. The maximum absolute estimation error across codes, the estimation error for INLmax, and the estimation error for

Table II Measurement results of the 12b DAC
(a) INL estimation errors in LSBs

	Max(abs(INL_est-INL_true))	(INLmax_est-INLmax_true)	(INLmin_est-INLmin_true)
ER	0.52	-0.056	-0.092
ER+I	0.3	-0.06	-0.047
uSMILE	0.37	-0.068	-0.065
uISMILE	0.34	-0.172	-0.109

(b) DNL estimation errors in LSBs

	Max(abs(DNL_est-DNL_true))	(DNLmax_est-DNLmax_true)	(DNLmin_est-DNLmin_true)
ER	0.6	0.133	-0.354
ER+I	0.16	-0.016	-0.01
uSMILE	0.42	0.242	-0.17
uISMILE	0.11	-0.025	0.016

INLmin have been summarized in Table II (a). Although the INL estimation accuracies look similar for all the 4 methods, there is a clear difference in the DNL estimation curves shown in Fig 10. Similar to what was noticed in the simulation results, for the algorithms which do not account for interpolation (ER and uSMILE), high DNL estimation errors can be seen at the MSB points. The summary for DNL estimation in Table I (b) also clearly shows that ER+I and uISMILE give significantly more accurate results for DNL. Since the specifications for DNL (commonly +/-1 LSB) are far more stringent than INL specifications, ER and uSMILE should not be used for linearity testing of Interpolated DACs.

Theoretically, the test time reduction as compared to the conventional method with 240 hit per code should be 20x. In practice, the reduction in total test time, which comprises data acquisition + on-chip calculation was ~19.8x for the ER+I method and ~16x for uISMILE. For higher resolution DACs, the test time reduction will be even more significant.

VII. Discussion

The uSMILE, uISMILE, ER and ER+I methods altogether provide 4 different ways to significantly reduce the linearity test time for DACs. There are pros and cons for both classes of methods: uSMILE/uISMILE and ER/ER+I. In uSMILE/uISMILE, since we need to take measurements at all the codes, we have to wait for the output of the DAC to settle before capturing the data for every code. Conversely, since the ER/ER+I involve taking measurements at fewer codes with multiple measurements per code, we do not have to wait for the output to settle when taking multiple measurements at the same code. It should be noted, though, that the settling time will be slightly lower for uSMILE/uISMILE because we are only stepping up one code at a time.

Another advantage that ER/ER+I have over uSMILE/uISMILE is that the data processing is relatively very simple, and so, will be faster, and more memory efficient. In fact, if one is interested in knowing just the maximum and minimum INL/DNL, then it is not even necessary to store INLs/DNLs at all codes. The voltage output measurements at only a few codes need to be stored, and the outputs at all the other codes can be reconstructed and the INL calculated, one code at a time, while keeping track of just the max and min INL/DNL values. If the uSMILE/uISMILE methods are implemented as presented in the paper, they will take a very long time and lots of memory compared to ER/ER+I, because huge matrix inversions are involved. But, as mentioned earlier, there is a significant advantage that DAC uSMILE linearity testing offers over ADC uSMILE linearity testing, which we can utilize to reduce the time and memory requirements. This stems from the fact that, unlike for ADCs, we know exactly which code we are sending before we even capture the output data. Basically, the H matrix, which is dependent on the output codes for an ADC, is solely dependent on the input codes for a DAC, which are already known to us. Hence, the Least squares solution H_{inv} can be pre-computed and stored, which drastically reduces computational test time and memory requirements. In fact, if one observes the H_{inv} matrix carefully, one will find that there are patterns in it which repeat, and thus, an extremely low number of distinct pre-computed values need to be stored for the H_{inv} matrix in the memory. We still need to store the preliminary INLs for all the codes, though, and there is no way to avoid this. Although the net test time of uSMILE will usually be higher than for ER, if the H_{inv} matrix is pre-computed and stored, then this increase in test time is not very significant. If we have an ADC on-chip to measure the DAC output, it paves the way for a complete BIST solution. Both ER/ER+I and uSMILE/uISMILE have the potential to be implemented on-chip.

The uSMILE/uISMILE methods are not without their advantages over the ER/ER+I methods. One significant advantage is that since we are actually measuring the output for each and every code in uSMILE/uISMILE, they have the potential to catch sparkle INLs/DNLs if they happen to occur. This can be done by performing a simple sanity check on the preliminary noisy INLs. In ER/ER+I, since we only measure the output at a few specific codes, and not all codes, they will not be able to catch these kinds of errors. Another significant advantage of uSMILE/uISMILE is that they can be combined with the ROME method [15] in order to facilitate measurement of DAC nonlinearities with low-linearity digitizers in addition to reducing the number of samples to be taken. Therefore, both methods have their advantages and disadvantages and can be applied depending on the requirements.

VIII. Conclusion

Novel linearity testing methods that significantly reduce test time have been discussed for various DAC architectures. The uSMILE algorithm and the Extrapolated Reconstruction (ER) method have been shown to be very effective in reducing the linearity test time for different segmented DAC architectures. Shortcomings of the segmented model have been discussed for DACs with interpolation, and new methods that account for interpolation (uISMILE and ER+I) have been developed. For all the proposed methods, extensive simulation results of 16-bit DACs and measurement results for some 12-bit DACs show that they are effective in significantly reducing time for linearity testing by reducing the number of samples that need to be measured, while giving accurate estimations of the INL and DNL errors.

Acknowledgments

Materials presented in this paper are based upon work supported by Texas Instruments Inc. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the

views of the sponsors.

REFERENCES

[1] M. Burns, G. W. Roberts, and F. Taenzler, *An introduction to mixed-signal IC test and measurement*, vol. 2001. Oxford University Press New York, 2001.

[2] "IEEE Standard for Terminology and Test Methods of Digital-to-Analog Converter Devices," *IEEE Std 1658-2011*, pp. 1–126, Feb. 2012.

[3] Y. Zhuang, B. Magstadt, T. Chen, and D. Chen, "High-Purity Sine Wave Generation Using Nonlinear DAC With Predistortion Based on Low-Cost Accurate DAC-ADC Co-Testing," *IEEE Trans. Instrum. Meas.*, vol. PP, no. 99, pp. 1–9, 2017.

[4] L. Jin, K. Parthasarathy, T. Kuyel, D. Chen, and R. L. Geiger, "Accurate testing of analog-to-digital converters using low linearity signals with stimulus error identification and removal," *IEEE Trans. Instrum. Meas.*, vol. 54, no. 3, pp. 1188–1199, Jun. 2005.

[5] H. Xing, D. Chen, and R. Geiger, "On-chip at-speed linearity testing of high-resolution high-speed DACs using DDEM ADCs with dithering," in *2008 IEEE International Conference on Electro/Information Technology*, 2008, pp. 117–122.

[6] X. L. Huang and J. L. Huang, "ADC/DAC Loopback Linearity Testing by DAC Output Offsetting and Scaling," *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 19, no. 10, pp. 1765–1774, Oct. 2011.

[7] H.-W. Ting, S.-J. Chang, and S.-L. Huang, "A Design of Linearity Built-in Self-Test for Current-Steering DAC," *J. Electron. Test.*, vol. 27, no. 1, pp. 85–94, Feb. 2011.

[8] K. Arabi, B. Kaminska, and M. Sawan, "On chip testing data converters using static parameters," *IEEE Trans. Very Large Scale Integr. VLSI Syst.*, vol. 6, no. 3, pp. 409–419, Sep. 1998.

[9] I. H. S. Hassan, K. Arabi, and B. Kaminska, "Testing digital to analog converters based on oscillation-test strategy using sigma-delta modulation," in *Proceedings International Conference on Computer Design. VLSI in Computers and Processors (Cat. No.98CB36273)*, 1998, pp. 40–46.

[10] K. P. S. Rafeeque and V. Vasudevan, "A built-in-self-test scheme for digital to analog converters," in *17th International Conference on VLSI Design. Proceedings.*, 2004, pp. 1027–1032.

[11] J.-L. Huang, C.-K. Ong, and K.-T. Cheng, "A BIST scheme for on-chip ADC and DAC testing," in *Proceedings Design, Automation and Test in Europe Conference and Exhibition 2000 (Cat. No. PR00537)*, 2000, pp. 216–220.

[12] Z. Yu and D. Chen, "Algorithm for dramatically improved efficiency in ADC linearity test," 2012, pp. 1–10.

[13] T. Chen and D. Chen, "Ultrafast stimulus error removal algorithm for ADC linearity test," in *2015 IEEE 33rd VLSI Test Symposium (VTS)*, 2015, pp. 1–5.

[14] X. Jin *et al.*, "An on-chip ADC BIST solution and the BIST enabled calibration scheme," in *2017 IEEE International Test Conference (ITC)*, 2017, pp. 1–10.

[15] S. K. Chaganti, T. Chen, Y. Zhuang, and D. Chen, "Low-cost and accurate DAC linearity test with ultrafast Segmented Model Identification of Linearity Errors and Removal Of Measurement Errors (uSMILE-ROME)," in *2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, 2018.