Indirect Background Calibration of Pipelined ADCs via Decision Boundary Gap Estimation (DBGE)

Lane Brooks, Hae-Seung Lee, Gregory Wornell

October 18, 2005

Abstract—A method of indirect background digital calibration of the dominant static non-linearities in pipelined ADCs is presented. The method, called Decision Boundary Gap Estimation (DBGE), monitors the output of the ADC to estimate the size of missing code gaps that result at the decision boundaries of each stage. Missing code gaps result from such effects as capacitor mismatch, finite op-amp gain, comparator offset, and charge injection. DBGE does not require special calibration signals or additional analog hardware and can even reduce the performance requirements the analog circuitry. The calibration is performed using the input signal and thus requires that the input signal exercise the codes in the vicinity of the decision boundaries of each stage. If it does not exercise these codes, then calibration does not matter because the non-linearities will not appear in the output signal. DBGE is simple and amenable to VLSI and/or processor implementations even in older fabrication technologies. Simulation results indicate DBGE is highly accurate, robust, and adaptive even in the presence of parameter drift and thermal noise.

Index Terms—pipelined analog-to-digital converter (ADC), adaptive digital background calibration, capacitor mismatch, finite op-amp gain, static nonlinearity.

1 Introduction

Pipelined analog-to-digital converters (ADCs) can realize high throughput and high resolution simultaneously. CMOS switched-capacitor based implementations have been widely researched and used in industry. In the absence of trimming or calibration, these implementations typically suffer from static non-linearities that limit the resolution to 8 to 10 bits [5], [9], [13].

These non-linearities have spurned many circuit and calibration techniques for realizing much higher resolutions. Analog circuit techniques such as those in [12] and [14] use analog components in the signal path to generate higher linearity at the expense of conversion speed. Digital circuit calibration techniques, which realize the benefits of device scaling, have also been developed and can be categorized into foreground and background techniques. Foreground calibration, as demonstrated in [9] and [10], requires interrupting normal ADC operation for calibration. In contrast, background techniques operate calibration circuits continuously and transparently so that users do not see service interruption.

Background techniques can further be classified into direct and indirect methods. Direct methods measure the errors with calibration signals during hidden calibration time slots. A "skip-and-fill" approach is used in [14] where the input samples are interpolated during a hidden calibration phase. A queue-based approach is used in [3]. The drawback of these approaches is that they require redundant channels/stages and/or their overall accuracy is a function of the coverage of the calibration signal, which cannot follow the same path as the signal exactly.

Indirect methods of calibration overcome the calibration signal coverage issue by estimating the errors from the input signal itself without the use of calibration signals. In [5] the dominant non-linearities of pipelined ADCs are modeled and corrected using adaptive equalization techniques prevalent in digital communications. It requires an additional "slow-but-accurate" ADC for reference and uses a very complicated algorithm for correction. In [8] they note that when an input signal has a low-pass input histogram, the non-linearities

of the ADC will generate high-pass components in the output histogram. Thus they collect an output histogram, low-pass filter it, and generate a correction map from the raw histogram space into the smoothed histogram space. In [7] they also use code densities or histograms with a second ADC to generate a correction map. These techniques are to varying degrees either algorithmically or hardware intensive such that they are not VLSI and/or computationally efficient, even at state-of-art silicon device scales.

Indirect calibration requires making assumptions about the input signal and possibly the errors themselves. For example, [8] assumes the input signal distribution is low-pass and [6] requires the input signal be over-sampled so that the spare bandwidth can be used to estimate assumed non-linearities. Presented herein is a technique called Decision Boundary Gap Estimation (DBGE) for indirect digital background calibration. DBGE removes the dominant non-linearities of pipelined ADCs that appear as missing code gaps at decision boundaries. DBGE, therefore, models these gaps and relies on the input signal to exercise the codes in the neighborhood of these gaps to estimate and remove them.

2 Pipelined ADC Error Models

A pipelined ADC consists of lower resolution ADCs, as shown in Figure 1, concatenated together to form the desired resolution. Initially we consider a 1 bit/stage (bps) pipelined ADC and later extend the concept of DBGE to other resolutions. The typical 1 bps pipeline stage implementation is shown in Figure 2. The dominant static non-linearities of such a circuit implementation are caused by capacitor mismatch, finite op-amp gain, comparator offset, and charge injection [2], [5], [9], [13].

Each of these effects are modeled and characterized in Figure 3. The plots on the left show the analog voltage transfer function of a single 1 bps pipeline stage. The plots on the right show the resulting digital output when the remaining stages are linear. Each of these effects produces a non-linearity at the decision boundary of the comparator. When all these effects are considered together as in the last plot of Figure 3, the digital output contains a

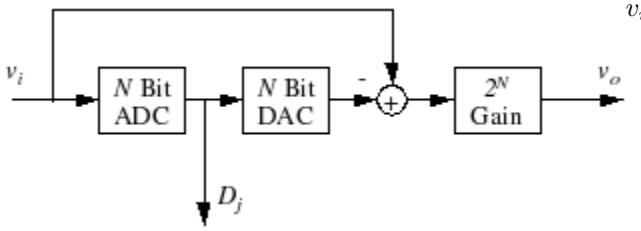


Figure 1: Block diagram of an N bps pipeline stage.

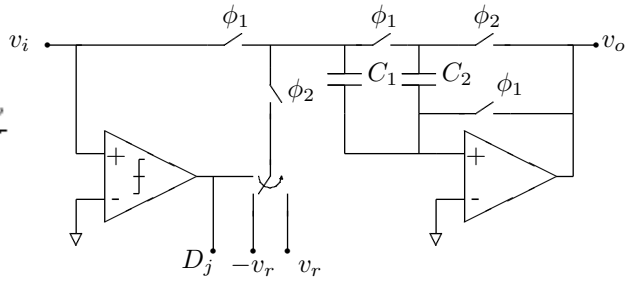


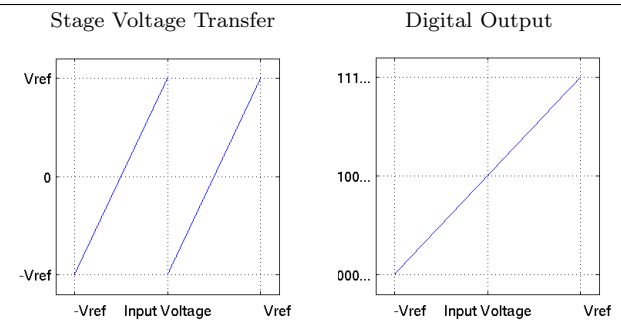
Figure 2: Typical circuit implementation of 1 bps pipeline stage. Single-ended version shown for simplicity.

single non-linearity of missing codes that straddles the decision boundary of the comparator. DBGE works by estimating the size of this missing code gap and removing it to make the ADC linear.

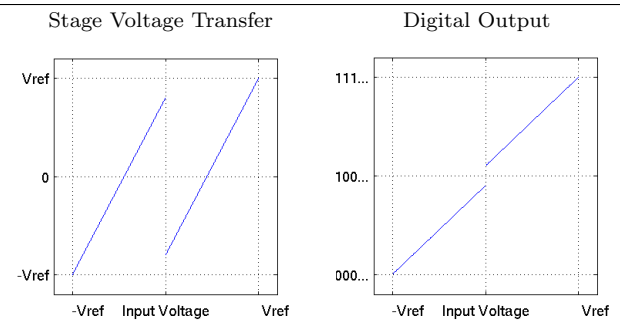
An important caveat to realize when designing for DBGE is that an intentional radix reduction is necessary to ensure that a missing code gap results even under worst case offsets and mismatch [9]. Thus for this 1 bps ADC, each stage requiring correction must have a radix less than 2. If this is not the case, then the residue amplification can go out of range causing a duplicate code region. DBGE cannot correct for duplicate code regions. The exception to this are designs with redundancy for over-range protection. Since they do not produce duplicate regions, they do not require a radix reduction.

3 Gap Correction

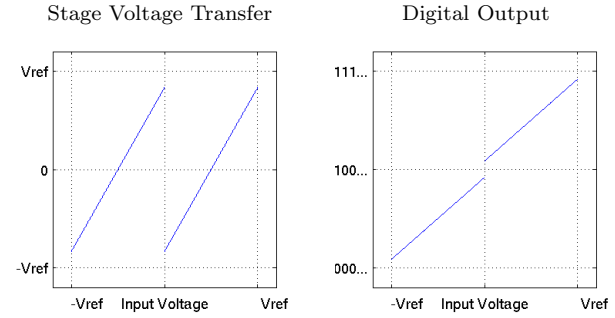
In a 1 bps pipelined ADC, the resolution of the ADC is set by the number of stages. If we start with a $k-1$ bit ADC and want to add 1 more bit of resolution, we simply add 1 more stage at the beginning of the concatenation and k bits are generated. Since is it possible to build a linear 8 to 10 bit ADC without trimming or calibration, DBGE starts by assuming that the we have a linear $k-1$ converter and that adding stage k does not increase the resolution because of the resulting decision boundary missing code gap. Thus stage k is the first stage requiring correction.



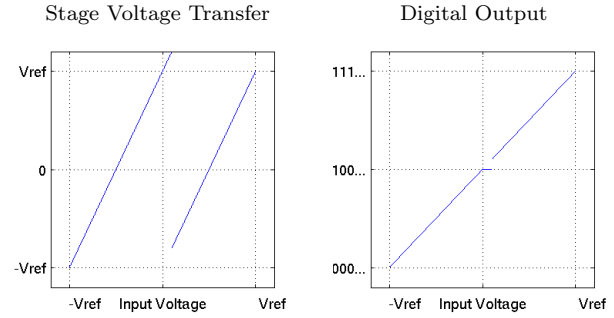
Ideal Case: $v_o = 2v_i + dv_{ref}$
$d = 1$ when the comparator output is high and $d = -1$ when it is low.



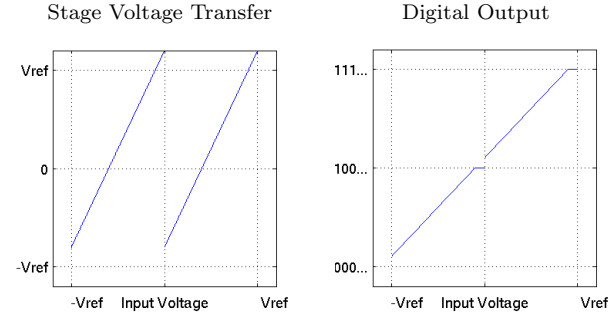
Capacitor Mismatch: $v_0 = (2 + \epsilon)v_i + (1 + \epsilon)dv_{ref}$
ϵ is the percentage of capacitor mismatch. When $\epsilon < 0$ as depicted, a missing code gap results at the decision boundary of the digital output.



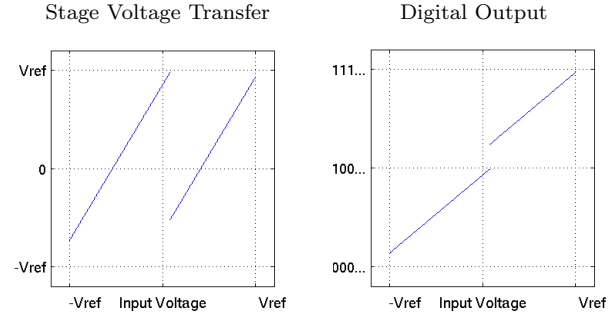
Finite Op-amp Gain: $v_o = \frac{2v_i + dv_{ref}}{1 + \frac{1}{A}}$
A is op-amp gain. When A is low enough it can produce a missing code gap in the digital output at the decision boundary.



Comparator Offset: $v_o = 2v_i + dv_{ref}$
Comparator offset shifts when d transitions from 1 to -1. Duplicate codes result on one side of the decision boundary and missing codes on the other side.



Charge Injection: $v_o = 2v_i + dv_{ref} + v_{os}$
v_{os} is the amount of offset added by charge injection. Up-down shifts in the voltage transfer function cause duplicate codes one side of the decision boundary and missing codes on the other side.



Everything: If the gain of the residue amplification is low enough, then when all errors are considered together the digital output contains a single non-linearity of missing codes that straddles the decision boundary.

Figure 3: Error models for various decision boundary non-linearities.

Correction of stage k proceeds as shown in Figure 4. When stage k produces a bit or decision output D_k, it is concatenated with the output of the other stages to produce the raw sample x_k. x_k is passed to the estimator to produce an estimate of the gap size.

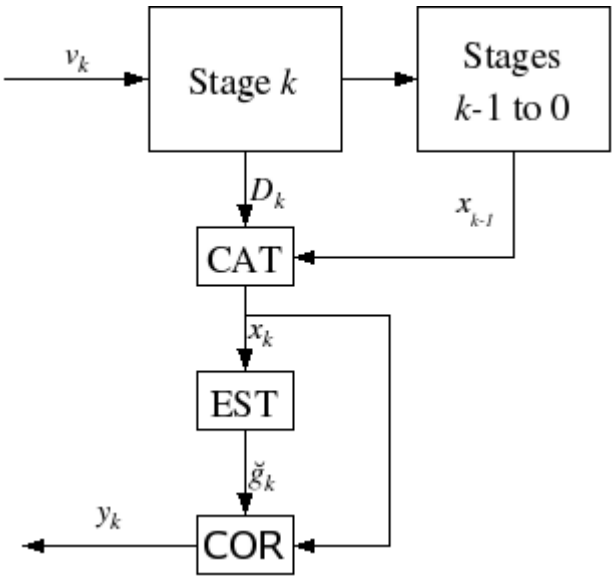


Figure 4: Block diagram of a single correction stage using DBGE. CAT block performs concatenation of bits. EST block produces an estimate of the decision boundary gap according to Eq 3. COR block corrects the sample according to Eq 1 to produce a linear output sample y_k.

Assuming the estimator produces a good estimate $\breve{g}_k$ of the gap size, then the non-linearity is removed from x_k by subtracting the gap size from all samples above the gap. Expressed mathematically, the linearized or corrected sample y_k is

$$y_k = \begin{cases} x_k, & \text{when} \quad D_k = 0 \\ x_k - \breve{g}_k, & \text{when} \quad D_k = 1 \end{cases} \tag{1}$$

Sample y_k is now free of the non-linearity that was limiting the overall resolution, and so an additional $k+1$ stage can be added by via concatenation. Stage $k+1$ can then be corrected in the same manner as stage k by using the corrected sample y_k. We will see later, however, that feeding the corrected sample into stage $k+1$ can cause problems with the estimator, so we will show in Section 4.3 how to concatenate further stages to avoid this problem.

This correction scheme has been demonstrated previously in [9]. There a sub-radix-2 pipelined ADC was designed and the gap was measured directly during a foreground calibration phase by driving the decision boundary voltage into each stage. This technique works well as witnessed by the 15 bit ADC. The drawback is that foreground calibration

requires taking the ADC out of service for calibration. Thus it suffers from calibration drift and/or service interruptions.

4 Gap Estimation

4.1 Initial Gap Estimate

In order to develop an adaptive and indirect decision boundary gap estimation, we start by assuming a linear $k-1$ bit ADC and add stage k to precede it. We model the resulting gap non-linearity produced by stage k with the signal flow graph of Figure 5. Here we do not model the effect of the unknown offset and gain, and initially we ignore all noise sources except the non-linearity induced at the decision boundary. The non-linearity is modeled as an unknown nonrandom parameter e_1 or e_0 added to the signal when the MSB decision D_k is 1 or 0 respectively. Our observation of the data is the digital output of the ADC x_k. The goal is to estimate the gap size g_k, which is $g_k = e_1 - e_0$. Since we constrain all stages needing correction to have a radix less than 2, e_0 must be less than 0 and e_1 must be greater than 0. In redundant designs such a 1.5 bps pipelined ADC, it is not necessary to reduce the radix because e_0, e_1, and g_k can be either positive or negative.

If we receive a collection of N samples of x_k and split it into two sets X_1 and X_0 where X_1 is the set of all samples with an MSB decision $D_k = 1$ and X_0 is the set of all samples

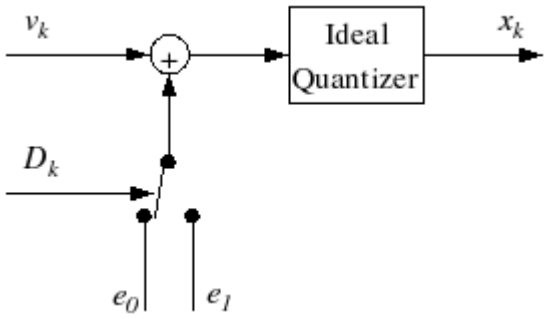


Figure 5: Signal flow graph showing how signal v_k gets corrupted by offsets e_0 and e_1 to produce bit boundary missing code gaps.

with $D_k = 0$, then we propose an initial estimate $\tilde{g}_k$ of the actual gap g_k to be

$$\begin{aligned} \tilde{e}_1 &= \min\{X_1\} \\ \tilde{e}_0 &= \max\{X_0\} \\ \tilde{g}_k &= \tilde{e}_1 - \tilde{e}_0. \end{aligned} \tag{2}$$

This estimator watches the data stream to find the maximum sample received below the decision boundary and minimum sample received above the decision boundary and subtracts the two to form the estimate $\tilde{g}_k$. Depending on the probability distribution of input voltage v_k, this estimate has varying degrees of performance. Whenever the probability distribution of v_k peaks or shares a peak at the decision boundary (which is midscale for a 1 bps ADC), then this estimate is a Maximum-Likelihood (ML) estimate. Qualitatively, the more likely the input signal is to exercise the codes at the decision boundary, the better this estimation performs and vice versa. This is a desirable trend given that the impact of the non-linearity is a function of how often the non-linearity appears in the data stream. Furthermore, if the input signal has finite probability to be within one quantum of the decision boundary, then as the number of samples approaches infinity, the bias of this estimate approaches 0. How quickly it converges depends on the probability density in the region of the decision boundary.

The above estimate has a very efficient processor and/or VLSI implementation. It requires 2 registers for tracking the minimum $\tilde{e}_1$ and maximum $\tilde{e}_0$ estimates and some comparison logic. Estimation proceeds as each sample is received. First the MSB decision D_k is checked. If it is 1, then the sample is compared against the minimum register and the minimum is updated if necessary. If D_k is 0, then the maximum register is compared and updated if necessary. To track changes in the gap that result from environmental changes, the minimum and maximum registers can be reset at a rate that matches the desired adaptation rate.

4.2 Improved Gap Estimate

The initial gap estimate provided in Equation 2 suffers from a problem when one includes the effects of additive thermal noise in the analog processing path. Figure 6 shows the results of a simulation which plots histograms of the X_1 set. In this simulation a gap $e_1 = 4.25$ was added to a uniformly distributed input source. The first histogram was created in the absence of thermal noise, and as expected, the $\tilde{e}_1$ estimate will correctly latch 4 as the minimum. Thus the gap will be removed to within the resolution of the ADC. In the second histogram, however, an additive white Gaussian noise with zero mean and standard deviation of 0.5 LSB was added to the source prior to quantization. The sharp edge of the histogram now appears smoothed, and the $\tilde{e}_1$ estimate will latch in 2 as an incorrect estimate. This shows how thermal noise biases the $\tilde{e}_0$ and $\tilde{e}_1$ estimates to under compensate for the actual gap size.

To deal with this one needs to reduce the thermal noise below the quantization noise. While this is desirable independent of DBGE, to avoid further constraining the thermal noise budget, we can effectively reduce the thermal noise below the quantization level with the following adjustments to the estimates:

$$\begin{aligned} \hat{e}_1 &= \tilde{e}_1 + s\left(1 - \frac{h_s[\tilde{e}_1]}{h_s[\tilde{e}_1 + s]}\right) \\ \hat{e}_0 &= \tilde{e}_0 - s\left(1 - \frac{h_{-s}[\tilde{e}_0]}{h_{-s}[\tilde{e}_0 - s]}\right) \\ \hat{g}_k &= \hat{e}_1 - \hat{e}_0. \end{aligned} \tag{3}$$

Here we define $h_s[m]$ as a super histogram bin made by summing s histogram bins according to

$$h_s[m] = \sum_{i=m}^{m+s-1} h[i]$$

where $h[i]$ is the histogram count of sample i. The motivation for these corrective terms can be seen in the last histogram of Figure 6. These correction terms groups s histogram bins

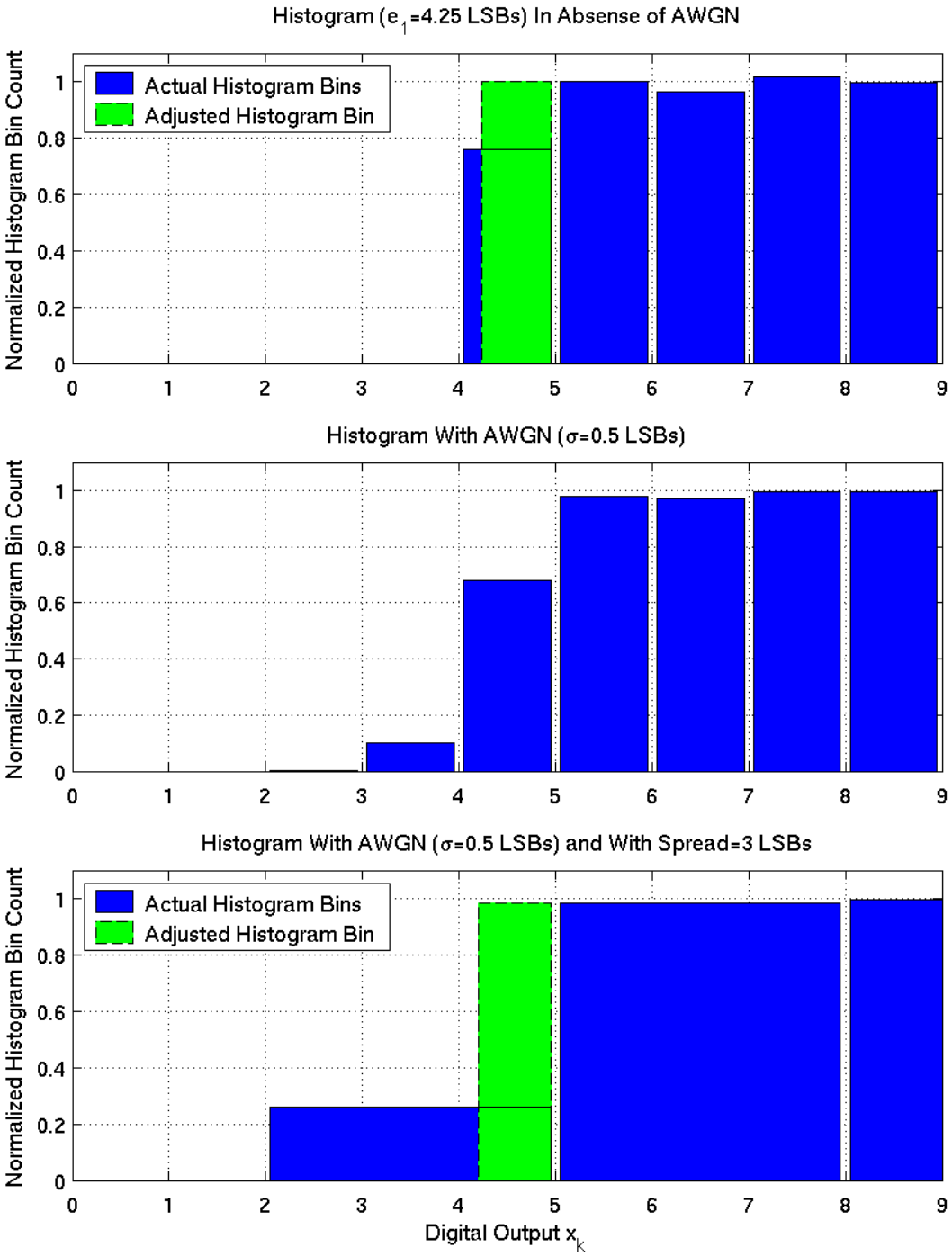


Figure 6: Histogram at bit boundaries a) in absence of noise, b) in presence of noise, and c) in presence of noise while using bin spreading to generate the adjusted minimum estimate.

together to form a super bin that spans the spread of the edge in the histogram caused by the thermal noise. We then adjust this bin to have the same height as his neighboring super bin and use the new width to indicate where the actual minimum is. Thus we approximate the input voltage that spans these two super bins as uniformly distributed. This approximation is reasonable for many applications, especially high bit depth ADCs.

If the number of samples used per estimate is large enough to approximate with certainty that $\tilde{e}_0$ and $\tilde{e}_1$ have latched in over compensating values, then using the uniform distribution approximation for the two adjacent super bins, the variance λ of the $\hat{g}_k$ estimate is bounded by $\lambda \leq \frac{2}{M_1} + \frac{2}{M_0}$. M_1 and M_0 are the number of samples collected in super bins $h_s[\hat{e}_1 + s]$ and $h_{-s}[\hat{e}_0 - s]$ respectively. This bound can be calculated by modeling each histogram bin as a binomial process under the constraint that M_1 and M_0 samples were received in the appropriate super bin. This bound has been verified in simulation, and it shows that as the spread s of the super bins increases, the ability to suppress the noise increases because M_0 and M_1 increase. The approximation that these bins are uniformly distributed over this spread may weaken as you increase the spread, so the spread should be designed to be as small as possible after the thermal noise level has been established.

This more robust estimate $\hat{g}_k$ is still very computationally friendly. Each estimate $\hat{e}_0$ and $\hat{e}_1$ requires an additional two registers for cumulating two super histogram bins. A division of these two registers must be performed, but since the estimate will be running at a very slow rate compared to that of the ADC, it can implemented serially using shifts and subtractions for minimal gate count.

4.3 Concatenation of Additional Stages

There are two different approaches that can be taken when concatenating additional stages in front of the stage k. One can either feed the raw sample x_k or the linearized sample y_k to the newly added stage $k+1$. If the linearized sample y_k is sent, then problems occur with the estimator of stage $k+1$ every time stage k updates his estimate. For example, if stage k

starts with a gap estimate of 0 and after some time updates that estimate to 10, the samples being fed to stage $k+1$ will suddenly change by 10. This skew between the samples already collected and the new samples will cause the estimator in stage $k+1$ problems.

One way to deal with this is to require a block based training procedure. This is only practical when DBGE is not being performed in real time but instead on a block of already collected samples. Under block based training, we start by training the estimator of stage k by passing the complete data set through it. We then pass the data through stage k again but this time training is disable and correction is enabled. Since stage k is done training, its estimates are stable and will not change, so the complete data set is corrected with a single gap estimate. This corrected block of data can then be sent to stage $k+1$. Again we feed the data set to stage $k+1$ in two phases—first for training and then for correction. This two phase correction scheme can then continue until all stages are trained and corrected.

If adaptive and real-time correction is desired, then a simple way to deal with the problem of lower order stages effecting the statistics of higher order stages is to feed the raw samples x_k to the estimators and let the corrected sample accumulated in parallel to it. When one does this, however, extra book keeping of the gap estimate for each stage is required because the corrected sample and the estimated gap size are done on different signals. The extra book keeping is quite minimal and requires a correction factor to each stage's gap estimate as follows

$$\begin{aligned} \breve{g}_n &= \hat{g}_n + \sum_{i=k}^{n-1} \breve{g}_i \\ &= \hat{g}_n - \hat{g}_{n-1} + 2\breve{g}_{n-1}. \end{aligned} \tag{4}$$

The resulting block diagram is shown in Figure 7, which shows the concatenation of 4 correction stages. For VLSI implementations, the logic required to concatenate stages is only 2 adders per stage.

Correction stages can be added to any desired bit depth. As stages are concatenated, it is important to realize how the input voltage probability distribution changes for each stage.

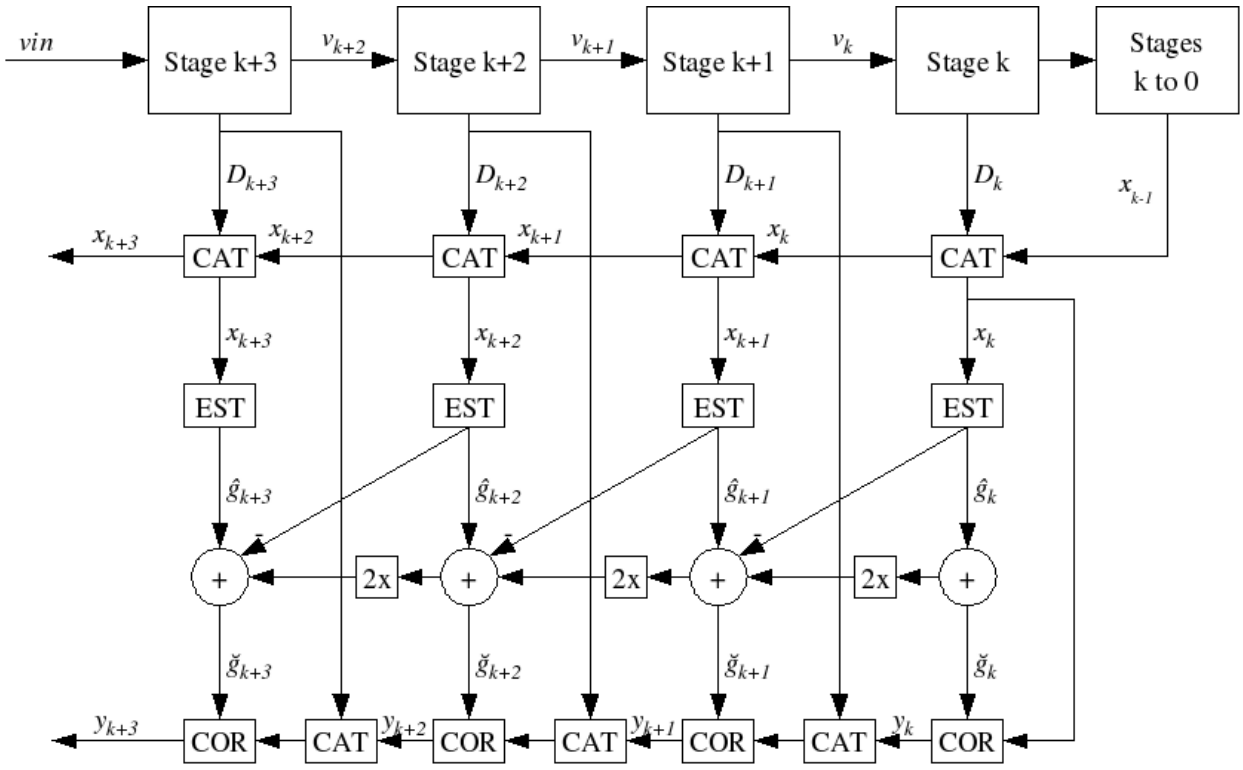


Figure 7: Block diagram of concatenated stages utilizing DBGE. CAT block performs concatenation of bits. EST block produces an estimate of the decision boundary gap according to Eq 3. COR block corrects sample according to Eq 1. Correction stages can be added *ad infinitum*.

The input voltage into the most significant stage is exactly that of the input signal. This stage takes the input signal, gains it by 2, cuts it in half, shifts the two segments on top of each other, and sums them. The effect on the probability distribution is shown in Figure 8 for several different input distributions over the first 3 stages. The dotted line of each graph is the decision boundary location where we approximate uniform probability density. One can see that the mixing incurred at each stage tends to flatten the probability density out and thus another validation of the approximation used in Equation 3.

5 Extensions

The motivation for DBGE came from the observation that the non-linearities that dominate CMOS switch-capacitor circuit design cause missing code gaps at each decision boundary.

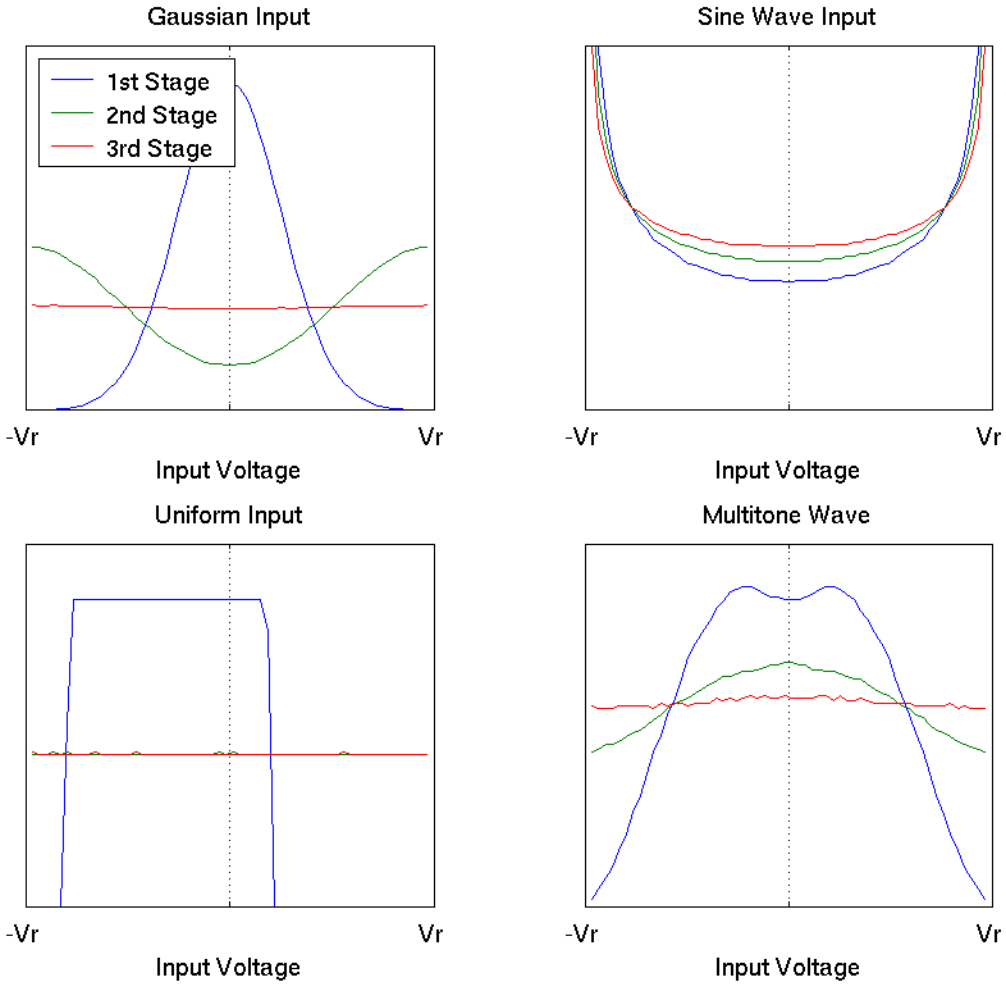


Figure 8: Input voltage probability distributions get mixed by each pipeline stage. Examples of a Gaussian, sine wave, and uniform input are shown. The dotted line shows the decision boundary location.

The technique, however, is general to a broader class of both implementations and architectures. It applies to any situation where the amplified error or residue from each stage causes a decision boundary gap. For the pipelined ADC this occurs when the pipeline stage divides the signal up and does not correctly map these signals directly on top each other as in the ideal case.

Thus, pipeline ADCs with multiple bits per stage can also benefit from DBGE. If redundancy is used as in a 1.5 bps ADC [11], the radix of each stage does not need lowered because the gap can go negative without producing duplicate codes. The reason that the

radix needs lowered for the 1 bps ADC is that the gap cannot go negative but gets clamped at 0 and produces duplicate codes. This makes the 1.5 bps ADC particularly well suited for DBGE. Higher resolution stages without over-range redundancy will need their radix lowered to make DBGE possible.

6 Simulation Results

DBGE has been simulated under several different conditions [4]. Shown here are the results of a 13 stage 1.5 bps pipelined ADC simulated with the mismatch parameters specified in Table 1. A 13 stage 1.5 bps ADC should produce 14 bits of resolution, but as shown in the INL and DFT plots of Figures 9 and 10, these mismatch parameters result in an effective 9 bit ADC.

Stage	Capacitor Mismatch	Op-amp Gain	Comparator Offset	Voltage Offset
	%	-	% of V_{ref}	% of V_{ref}
12	1.19	342	0.24	-0.41
11	1.64	106	-0.06	-0.30
10	-1.21	197	4.72	0.16
09	-0.65	154	-2.07	0.39
08	1.07	421	2.71	-0.15
07	-0.88	162	0.26	-0.43
06	0.21	260	2.69	-0.48
05	-0.09	151	-0.99	-0.04
04	0.51	143	3.91	-0.43
03	-0.54	499	-2.16	-0.26
02	0.05	187	-1.47	0.47
01	1.80	305	3.07	0.40
00	1.66	435	4.19	0.35

Table 1: Simulation Setup Consisted of 13 1.5 bps stages.

For these simulations the effects of thermal noise are modeled by adding a dose of zero mean additive white Gaussian noise with a standard deviation of 0.22 LSBs to each stage for each sample. Thus, the total thermal noise at the output is 0.25 LSBs. DBGE was then performed on the first 7 stages. 100,000 zero mean Gaussian samples with standard

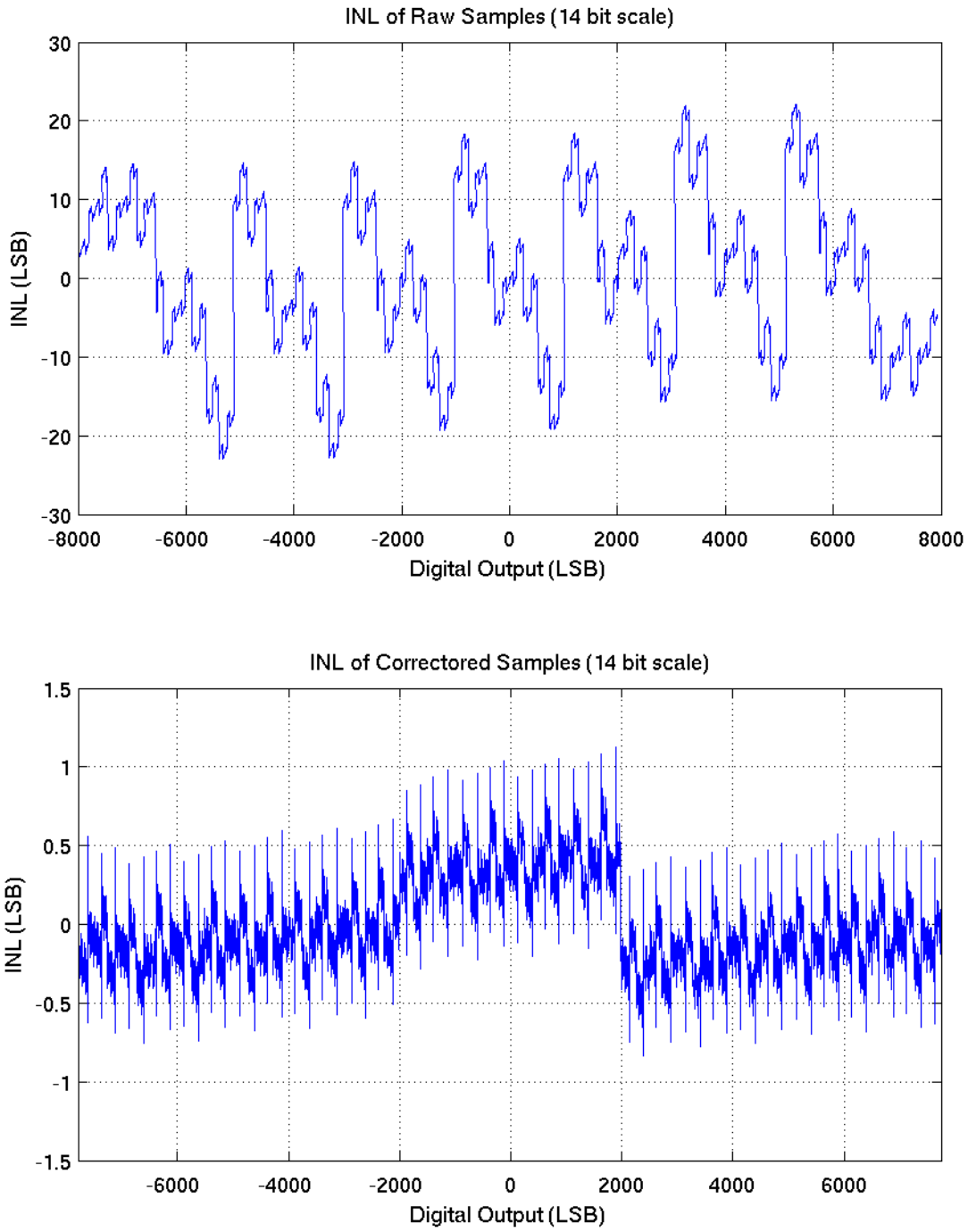


Figure 9: INL of 13 stage 1.5 bps ADC with mismatch parameters specified in Table 1. Uncorrected ADC has INL as big as ± 20 LSBs on a 14 bit scale. After DBGE, the resulting INL is ± 1 LSB.

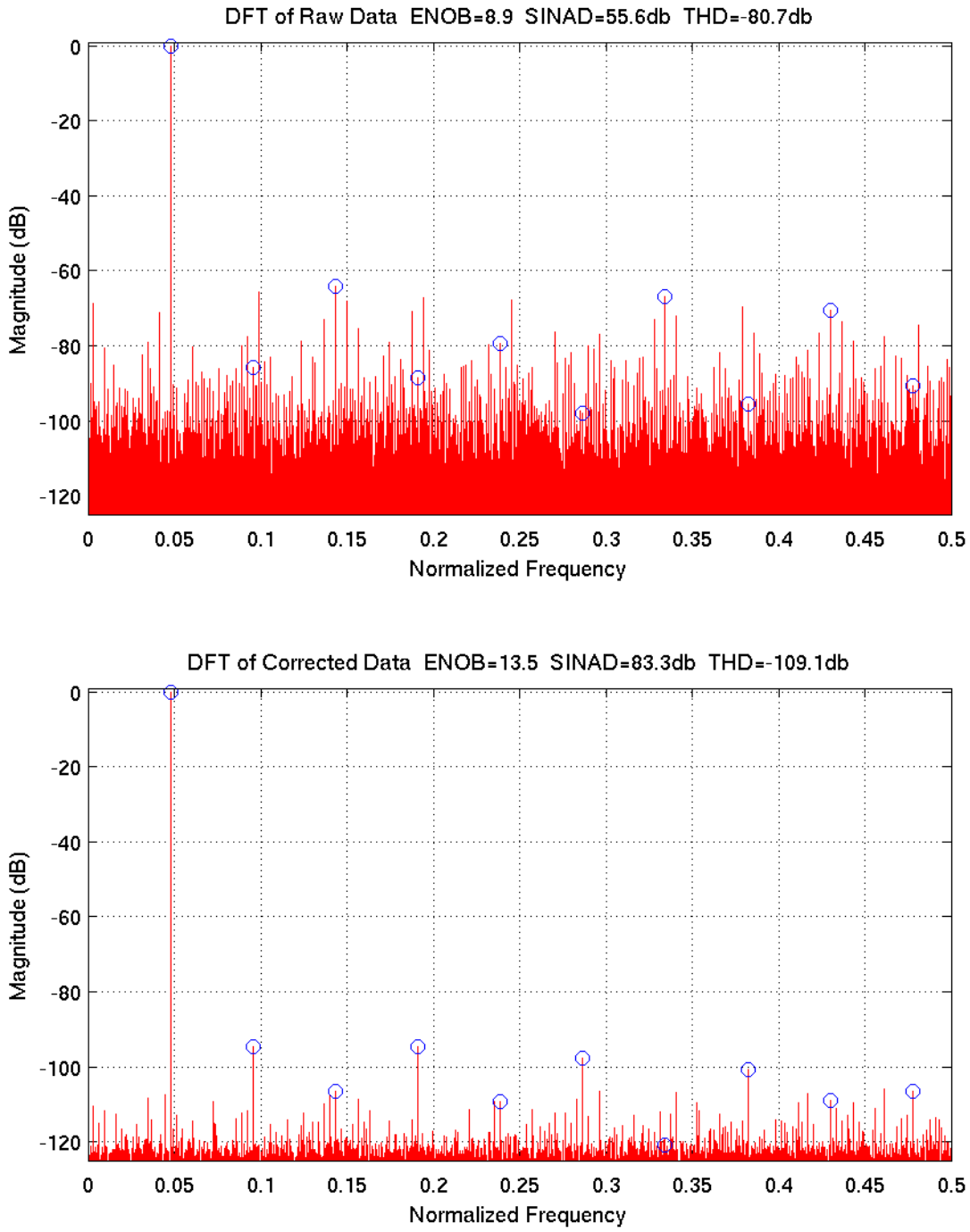


Figure 10: DFT of 13 stage 1.5 bps ADC with mismatch parameters specified in Table 1 resulting from a full scale sine wave. Uncorrected ADC has ENOB of 9 bits. After DBGE, the resulting ENOB is 13.5 bits.

deviation of $\frac{V_{ref}}{5.5}$ were sent into the ADC for training. After DBGE, the resulting effective number of bits is 13.5 (see Figures 9 and 10), which is only 0.5 bits lower than the same ADC under ideal conditions. The ENOB (Effective Number of Bits), SINAD (Signal to Noise and Distortion Ratio), and THD (Total Harmonic Distortion) were measured according to the procedures in [1]. Table 2 summarizes the results for both the raw and corrected ADC.

	ENOB (bits)	SINAD (db)	THD (db)
Raw	8.9	55.6	-80.7
Corrected	13.5	83.3	-109.1

Table 2: Simulation Results.

7 Conclusion

Decision Boundary Gap Estimation (DBGE) is an adaptive, digital, indirect method of background calibration for removing missing code gaps at decision boundaries in ADCs. These are the dominant static non-linearities in pipelined ADCs, and DBGE provides a stochastic approach to removing them. The performance, therefore, is directly related to likelihood of the input signal to be in the neighborhood of these decision boundaries, and simulation results have verified the performance over a wide range of signals and conditions.

The advantages of DBGE are numerous. There is no need for either additional analog hardware, such as a redundant channels/stages or a reference converter to calibrate against. The calibration is highly accurate because the transition points are directly aligned. Furthermore, its simplicity makes it very amenable to VLSI and/or processor based implementations. Thus, DBGE is a calibration approach that can be implemented to improve existing ADC designs or to shape new designs by relaxing analog circuit requirements for high gain op-amps, matched capacitors, and low offset comparators. Reducing these design constraints allows the designer to reduce power and/or increase conversion speed, and perhaps most importantly, it can be an enabling factor for ADC design in deep sub-micron technologies.

References

[1] IEEE standard for terminology and test methods for analog-to-digital converters. *IEEE STD 1241-2000*, December 2000.

[2] Eulalia Balestrieri, Pasquale Daponte, and Sergio Rapuano. A state of the art on ADC error compensation methods. *IEEE Transactions on Instrumentation and Measurement*, pages 1388–1394, August 2005.

[3] E.B. Blecker, T.M. McDonald, O.E. Erdogan, P.J. Hurst, and S.H. Lewis. Digital background calibration of an algorithmic analog-to-digital converter using a simplified queue. *IEEE Journal of Solid State Circuits*, pages 1489–1497, June 2003.

[4] Lane Brooks. http://www.brooks.nu/∼lane/pubs.

[5] Yun Chiu, Cheongyuen W. Tsang, Borivoje Nikolic, and Paul R. Gray. Least mean square adaptive digital background calibration of pipelined analog-to-digital converter. *IEEE Transactions on Circuits and Systems*, pages 38–46, January 2004.

[6] Vijay Divi and Greg Wornell. Signal recovery in time-interleaved analog-to-digital converters. *IEEE ICASSP*, 2004.

[7] Udaykiran Eduri and Franco Maloberti. Online calibration of a nyquist-rate analog-to-digital converter using output code-density histograms. *IEEE Transactions on Circuits and Systems*, pages 15–24, January 2004.

[8] J. Elbornsson and J.-E. Eklund. Histogram based correction of matching errors in subranged ADC. *ESSCIRC 2001*, pages 555–558, September 2001.

[9] Andrew Karanicolas, Hae-Seung Lee, and Kantilal Bacrania. A 15-b 1-msample/s digitally self-calibrated pipeline ADC. *IEEE Journal of Solid-State Circuits*, pages 1207–1215, December 1993.

[10] Hae-Seung Lee, D.A Hodges, and P.R. Gray. A self-calibrating 15 bit cmos A/D converter. *IEEE Journal of Solid State Circuits*, pages 813–819, December 1984.

[11] Stephen Lewis, Scott Fetterman, George Gross, R. Ramachandran, and T.R. Viswanathan. A 10-b 20-msample/s analog-to-digital converter. *IEEE Journal of Solid State Circuits*, pages 351–358, March 1992.

[12] P.W. Li, M.J. Chin, P.R. Gray, and R. Castello. A ratio-independent algorithmic analog-to-digital conversion technique. *IEEE Jounral of Solid State Circuits*, pages 828–836, December 1984.

[13] U.-K. Moon and B.-S. Song. Background digital calibration techniques for pipelined ADC's. *IEEE Transactions on Circuits and Systems*, pages 102–109, February 1997.

[14] B.-S. Song, M.F. Tompsett, and K.R. Lakshmikumar. A 12-bit 1-msample/s capacitor error-averaging pipelined A/D converter. *IEEE Journal of Solid-State Circuits*, pages 1316–1323, December 1988.