A Fast Simulator for Pipelined A/D Converters

Bibhu Datta Sahoo and Behzad Razavi
Electrical Engineering Department
University of California, Los Angeles
e-mail:bsahoo@ee.ucla.edu,razavi@ee.ucla.edu

***Abstract*—A simulator for comprehensive analysis of pipelined A/D converters has been developed in MATLAB and compiled to an executable that can be run on various platforms. The simulator accepts user-specified parameters such as resolution per stage, number of stages, input and clock frequencies, input amplitude, op amp nonlinearity, capacitor mismatch, comparator offset, and clock jitter. The tool then provides the simulated performance in the form of residue plots, differential and integral nonlinearity profiles, output spectrum, and input-referred thermal noise. Compared with Cadence's Spectre, the proposed simulator runs several orders of magnitude faster while incurring a small error.**

I. INTRODUCTION

The design of high-performance analog-to-digital converters (ADCs) presents difficult challenges as applications call for higher speeds and resolutions and as device dimensions and supply voltages are scaled down. To meet these challenges, increasingly sophisticated ADC architectures, circuit techniques, and digital processing and calibration algorithms have been introduced, leading to complexity levels of tens of thousands of transistors and extremely lengthy analog simulations. These issues are particularly problematic in pipelined ADC architectures because the imperfections of the stages tend to accumulate, thus complicating the analysis of the overall performance.

This paper introduces a simulation tool for the analysis and design of pipelined ADCs that runs orders of magnitude faster then circuit simulators. Based on a MATLAB script but usable without MATLAB, the tool operates through a graphical user interface (GUI) and provides differential nonlinearity (DNL) and integral nonlinearity (INL) profiles, the input-output characteristic, the input-referred thermal noise resulting from op amp noise and kT/C noise, and the output spectrum in response to an input sinusoid. The tool incorporates critical imperfections such as op amp nonlinearity, capacitor mismatch, op amp slewing and linear settling, comparator offset, and clock jitter.

Section II of the paper deals with the modeling of pipelined ADCs and their nonidealities in a platform such as MATLAB, including a state-space methodology for the computation of noise in pipelined ADCs. Section III presents the simulator engine, Section IV describes the simulator interface, and Section V compares simulation results with those obtained from Cadence.

II. PIPELINED ADC MODELING

Figure 1 shows the architecture of a pipelined ADC. Each stage digitizes its analog input, V_{in}, by means of a sub-ADC with a resolution of M bits (not necessarily the same for all stages), converts this "estimate" to digital by means of a sub-digital-to-analog converter (sub-DAC), and subtracts the analog estimate from V_{in} to produce a "residue." The residue is amplified by a power of two and passed on to the next stage for further digitization.

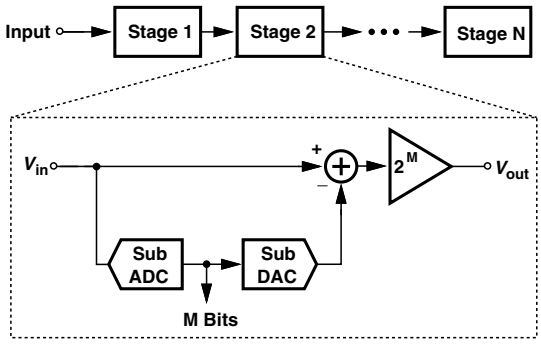


Fig. 1. Pipelined ADC architecture.

A compact, efficient implementation of pipelined stages is shown in Fig. 2. Called a "1.5-bit multiplying DAC" (MDAC) stage [1], [2], this circuit employs only two comparators in the sub-ADC to determine whether V_{in} lies below $-V_{REF}/4$, between $-V_{REF}/4$ and $+V_{REF}/4$, or above $+V_{REF}/4$. The sub-ADC and capacitors C_1 and C_2 sample V_{in} simultaneously, C_2 is flipped around the op amp, and the sub-ADC's decision determines whether the left plate of C_1 must be switched to $-V_{REF}$, 0, or $+V_{REF}$. The popularity of this architecture stems from its tolerance of large comparator offsets and timing mismatches between the sampling paths of $C_1 + C_2$ and the two comparators.

The design of the above pipelined stage must deal with the following imperfections: (1) kT/C noise given by C_1 and C_2 in both sampling and amplification modes, (2) op amp noise in the amplification mode (and, in some topologies, the sampling mode), (3) finite open-loop gain of the op amp, (4) nonlinearity of the op amp, (5) mismatch between C_1 and C_2, (6) input capacitance of the op amp, C_x, (7) slewing and linear settling of the op amp in the amplification mode while it drives C_L (the input capacitance of the next stage in the pipeline), and (8) the offset of the comparators. In the front-end stage of the ADC, the clock jitter also plays a critical

978-1-4244-4480-9/09/$25.00 ©2009 IEEE

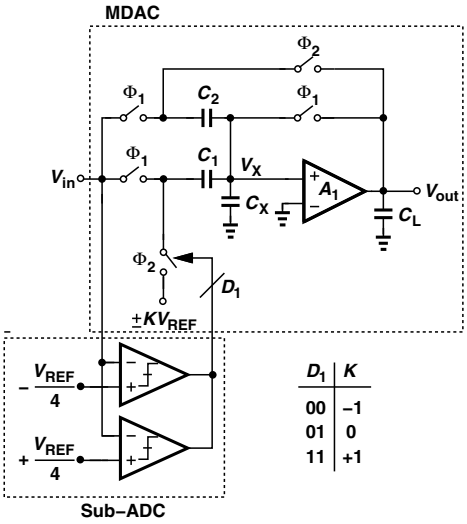


Fig. 2. Pipelined stage with 1.5-bit resolution.

role.

A. Modeling of Static Effects

We now formulate the behavior of the stage so as to capture the above nonidealities. The input-output static characteristic of the op amp in Fig. 2 can be approximated by a third-order polynomial across the voltage range of interest:

$$V_{out} \approx \alpha_1 V_x + \alpha_2 V_x^2 + \alpha_3 V_x^3. \quad (1)$$

In the sampling mode, the input and output voltages of the op amp are equal and expressed by

$$V_{x1} = \alpha_1 V_{x1} + \alpha_2 V_{x1}^2 + \alpha_3 V_{x1}^3, \quad (2)$$

where V_{x1} denotes V_x in the sampling mode. (The offset of the op amp can also be included by subtracting it from V_{x1} on the right-hand side.) The total charge stored on C_1 and C_2 is thus equal to

$$Q_{samp} = (C_1 + C_2)V_{in} - (C_1 + C_2 + C_x)V_{x1}. \quad (3)$$

Upon completion of sampling, the sub-ADC is clocked, C_2 is placed around the op amp, and the left plate of C_1 is connected to KV_{REF}, where $K = \pm 1, 0$ denotes the sub-ADC's decision. The output voltage thus settles to

$$V_{out} = \alpha_1 V_{x2} + \alpha_2 V_{x2}^2 + \alpha_3 V_{x2}^3, \quad (4)$$

where V_{x2} is the value of V_x in the amplification mode. Also the total charge residing on C_1, C_2 and C_x in this mode is given by

$$Q_{amp} = C_1 K V_{ref} + C_2 V_{out} - (C_1 + C_2 + C_x)V_{x2}. \quad (5)$$

Conservation of charge requires that $Q_{amp} = Q_{samp}$. Equations (2)-(5) provide a complete description of the static behavior of the pipelined stage, modeling the following nonidealities: finite op amp gain, op amp nonlinearity, op amp offset, op amp input capacitance, and mismatch between C_1 and C_2.

Equations (2)-(5) do not lend themselves to a closed-form solution but are readily solved in MATLAB using the function "*roots*". That is, the output of each stage in the pipeline is calculated based on the user-specified nonidealities of the stage and passed on to the next stage.

The coefficients α_j in Eq. (1) are obtained by transistor-level simulation of the op amp and fitting a third-order polynomial to the resulting characteristic. For differential implementations with small mismatches, $\alpha_2 \approx 0$, thus simplifying the fitting procedure.

The offset of the comparators is incorporated by adjusting their decision thresholds and hence producing a corresponding digital output for the sub-ADC. The offsets can be specified by the user for all stages.

While the above modeling procedure has been derived for a 1.5-bit stage, our simulation tool in fact handles arbitrary resolutions in each stage of the pipeline. Equations (2)-(5) can be modified to handle N-bit MDACs with or without redundancy.

B. Modeling of Dynamic Effects

In the analysis and design of pipelined ADCs, it is desirable to quantify how the precision degrades as the clock rate increases. The simulation tool developed here approximates the MDAC dynamic behavior by a slewing regime followed by a linear settling regime . The time at which the settling changes from nonlinear to linear (t_1) is a user-defined fraction of the clock period. The linear settling behavior is based on the small-signal model shown in Fig. 3, whose time constant is given by [3]

$$\tau_{amp} = \frac{C_L(C_S + C_x) + C_L C_F + (C_S + C_x)C_F}{G_m C_F}. \quad (6)$$

In other words, an exponential with this time constant and an initial value equal to $V_{out}(t_1)$ is solved to obtain V_{out} at $t = T_{ck}/2$.

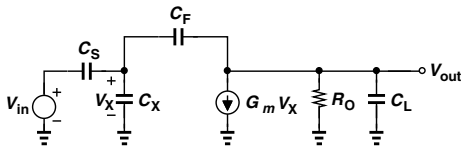


Fig. 3. Equivalent circuit of MDAC in amplification mode.

C. Modeling of Noise Transfer Functions

Perhaps the most difficult task in the analysis of pipelined ADCs, computation of the overall input-referred noise plays a critical role in the power optimization of the design. Since such ADCs typically scale down the devices and bias currents along the pipeline, it is essential to accurately calculate the noise contribution of each stage.

The difficulty in noise computation arises from two factors: (1) the noise transfer functions become very complex, assuming orders as high as 6 or 7 (explained below), and (2) the noise sampled by the input of one stage depends on

bandwidth limitations imposed by the MDAC in the previous stage. Cadence's Spectre can calculate noise in transistor-level transient simulations, but the simulation becomes prohibitively long for a cascade of pipelined stages and, owing to difficulty in finding periodicity, fails in some cases. In this work, we apply the state space method of analyzing linear systems [4] to the calculation of the noise.

Figure 4 shows the small-signal model of an MDAC employing a two-stage op amp—a common choice in today's low-voltage ADCs. Here, R_{sj} and V_{sj} respectively denote the on-resistance and thermal noise of the switches that appear in series with the sampling capacitors, and $I_{n1,op}$ and $I_{n2,op}$ the noise current at the output of each stage of the op amp (per unit bandwidth). The op amp is modeled by two gain stages and a compensation capacitor C_c. Note that the input sampling network of the next stage is also included. The objective is to determine the total noise sampled on C_L at the end of this mode.

Rather than directly compute the noise transfer functions, we utilize the state space method: we first assign independent states to the storage elements in the circuit (voltages X_j in Fig. 4) and then write the following matrix equations:

$$\dot{\boldsymbol{X}} = \boldsymbol{AX} + \boldsymbol{BU} \quad (7)$$

$$\boldsymbol{Y} = \boldsymbol{CX} + \boldsymbol{DU}, \quad (8)$$

where $\boldsymbol{X}$ denotes the states, $\dot{\boldsymbol{X}} = \boldsymbol{dX/dt}$, $\boldsymbol{U}$ is the vector of inputs to the system (the noise sources), and $\boldsymbol{Y}$ is the vector of the outputs (in this case, the voltage across C_L). Also, $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, and $\boldsymbol{D}$ are the "state matrices".

The circuit of Fig. 4 contains six states, five inputs and one output. Since the output is equal to state X_6, we have $\boldsymbol{D} = 0$ and $\boldsymbol{C} = [0\ 0\ 0\ 0\ 0\ 1]$. Matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ are defined by the circuit parameters and can be readily found by writing the various nodal equations. For example, the first row of $\boldsymbol{A}$, i.e., $a_{11}, ..., a_{16}$, is obtained by noting that

$$\dot{X_1} = a_{11}X_1 + \cdots + a_{16}X_6 + b_{11}V_{n1} + \cdots + b_{15}I_{n1,op} + b_{16}I_{n2,op}, \quad (9)$$

where b_{1j} are the first row of $\boldsymbol{B}$, V_{nj} are the four noise sources, and $I_{nj,op}$ the op amp noise currents. To compute a_{11}, we apply a KVL in the loop consisting of C_1, C_X, R_{s1}, and V_{ns1}, thus arriving at $a_{11} = -1/(R_{s1}C_1)$. With the four state matrices known, Eqs. (7) and (8) are numerically solved to find the contribution of each noise source to the output. These contributions are then integrated for a frequency range of 0 to ∞ and summed to determine the total noise. The numerical integration is simplified through the use of initial value theorem in Laplace transforms.

The circuit of Fig. 4 still does not completely represent the noise mechanisms in a cascade of pipelined stages. Consider the cascade depicted in Fig. 5, where the first stage is in the amplification mode and the second in the sampling mode. Here, the second op amp is configured as a unity-gain buffer by a feedback switch with noise $V_{nf1,2}$. Thus, the noise of the op amp and the feedback switch is sampled on $C_{1,2}$ and $C_{2,2}$ in the second stage along with the other noise sources included in the previous calculations. The state space approach can be extended to this case as well.

The foregoing noise calculation method continues to the end of the pipeline, thereby yielding the total integrated noise at the output of the last MDAC stage. The result is then divided by the total gain provided by the pipeline so as to arrive at the input-referred noise voltage of the ADC.

The circuits of Figs. 4 and 5 may imply that the value of the switch resistances must be provided by the user. Since these resistances should not limit the settling behavior of the MDAC, the tool assumes that the resistance in series with each capacitor leads to a time constant of $1/10$ of $T_{ck}/2$. Also, the op amp noise spectral density is approximated by $4kT\gamma g_m$ where γ denotes the excess noise factor of MOSFETs (≈ 1) and g_m denotes the transconductance of each stage of the op amp. As explained in Section V, these approximations lead to good agreement between our simulator's estimate and that obtained from Spectre.

III. Simulator Engine

The simulator engine is shown in Fig. 6. The arguments to the engine are the MDAC architecture, number of stages in the pipeline, clock frequency, input signal swing, input frequency and the variance of the clock jitter. The INL and DNL profiles are obtained by code density tests using either a ramp input or a sinusoidal input. The ramp input is chosen such that it produces 10 codes per bin. For sinusoidal inputs, the computation becomes difficult as the resolution exceeds 8 bits because hundreds of thousands of samples are necessary to produce the profiles. Thus, the simulation is performed in a number of passes, each generating 25,000 codes. The results are then combined properly, yielding the DNL and INL profiles.

In order to determine the output spectrum in response to an input sinusoid whose amplitude and frequency are defined by the user, the simulator first generates the digital bits from each stage, appropriately combines them to generate the digital representation of the analog value, and subsequently takes the FFT in MATLAB.

IV. Simulator Interface

The simulator has been implemented in MATLAB and subsequently compiled to an executable that can be run on various platforms. On the GUI, the user selects the resolution of each stage (1 bit, 1.5 bits, or higher integer values) and the number of stages in the pipeline. The user can also enter the following parameters: (1) for each stage, the op amp gain and nonlinearity coefficients, the capacitor values and mismatch, the op amp and comparator offsets, and the transconductance of the op amp; (2) the analog signal swing and frequency, and the clock frequency.

Upon simulation, the user can view the residue plot of each stage, the overall input-output characteristic, the DNL and INL profiles, and the output spectrum. The total input-referred noise along with contributions from each stage are also computed.

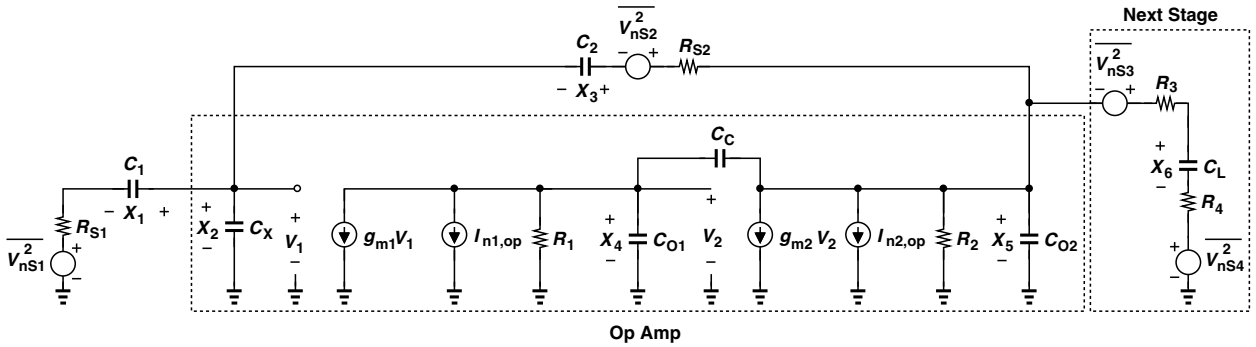


Fig. 4. MDAC equivalent circuit with all noise sources.

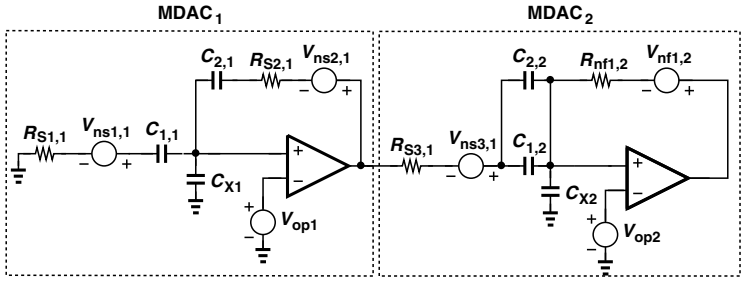


Fig. 5. Noise sources: stage 1 in amplification mode and stage 2 in sampling mode.

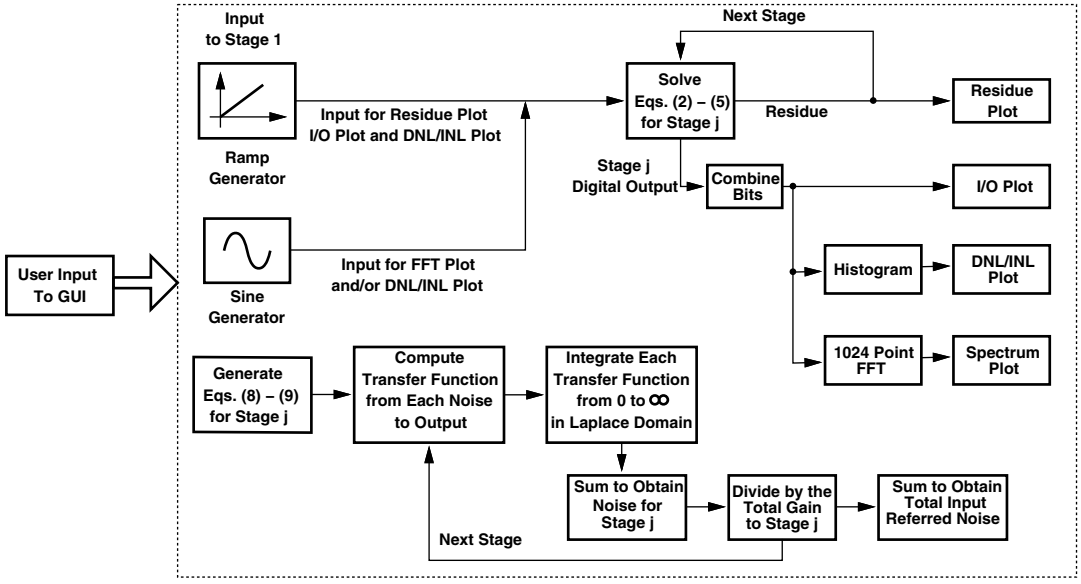


Fig. 6. Simulator engine.

The simulator also provides two calibration algorithms, either one of which can be selected and run on the ADC. Alternatively, the user can use his/her own calibration algorithm and compare the performance of the ADC before and after calibration. In the latter case, the user can write the digital outputs of the stages to a file, run the calibration algorithm, and return the results to the simulator so as to study the performance after calibration.

V. Simulation Results and Comparison with Spectre

In order to demonstrate the speed advantages of the proposed simulator, pipelined ADCs having resolutions of 8 to 12 bits have been simulated by the tool, by Spectre, and by Ultrasim on a 64-bit AMD, 2.2-GHz machine with Redhat Linux. Spectre and Ultrasim simulations were performed at transistor level (in 90-nm CMOS technology).

Table I compares the linearity simulation times using our

simulator and Ultrasim. (Spectre simulation was prohibitively long.) Figure 7 shows the DNL and INL plots obtained from the proposed tool and Ultrasim. Table II shows the

TABLE I
DNL/INL COMPUTATION TIME

Resolution	Ultrasim	Proposed Simulator
8-bit	5h:33m:40s	3.3s
10-bit	12h:45m:33s	45.97s
12-bit	63h:11m:24s	16m:37s

simulation time to compute the 1024-point FFT of the output for 8-bit, 10-bit and 12-bit ADCs. Table III compares the

TABLE II
1024 FFT COMPUTATION TIME

Resolution	Spectre	Ultrasim	Proposed Simulator
8-bit	17h:27m:12s	2h:2m:40s	1.24s
10-bit	20h:37m:19s	2h:26m:31s	1.63s
12-bit	36h:16m:22s	6h:43m:11s*	2.48s

*Accuracy tightened for 12-bit resolution.

simulation time and accuracy of noise measurement using the proposed tool and Spectre for stage 1 and stage 2 of a pipelined ADC and the cascade of the two stages. (Transient noise simulations for more than two pipelined stages in Spectre have not been successful.)

TABLE III
NOISE COMPUTATION

	Noise (μV)		Error	Simulation Time	
	Spectre	Proposed Simulator		Spectre	Proposed Simulator
Stage 1	110	120.70	9.7%	20m:10s	0.200s
Stage 2	197	208.99	6.1%	4m:36s	0.192s
Cascade	275	299	8.7%	26m:18s	0.3s

VI. CONCLUSION

A MATLAB-based simulator for the analysis and design of pipelined ADCs has been proposed that allows fast computation of critical parameters, enabling the user to easily explore the design space and quantify the consequences of each design choice. The simulator provides DNL and INL profiles, the output spectrum, and the input-referred noise in orders of magnitude less time.

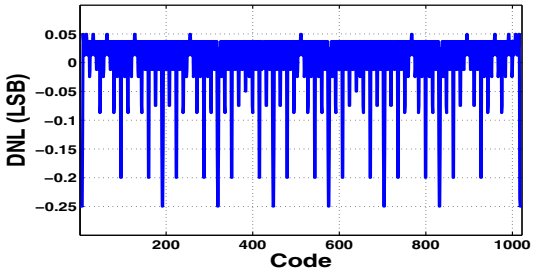
REFERENCES

[1] S. H. Lewis, "Optimizing the Stage Resolution in Pipelined, Multistage, Analog-to-Digital Converters for Video Rate Applications," *IEEE Trans. Circuits Syst. II*, vol. 39, pp. 516–523, Aug 1992.

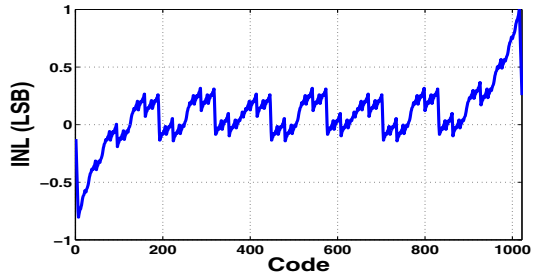
[2] A. M. Abo and P. R. Gray, "A 1.5-V, 10-bit, 14.3-MS/s CMOS Pipelined Analog-to-Digital Converter," *IEEE J. Solid-State Circuits*, vol. 34, pp. 599–606, May 1992.

[3] B. Razavi, *Design of Analog CMOS Integrated Circuits*. McGraw Hill, 2001.

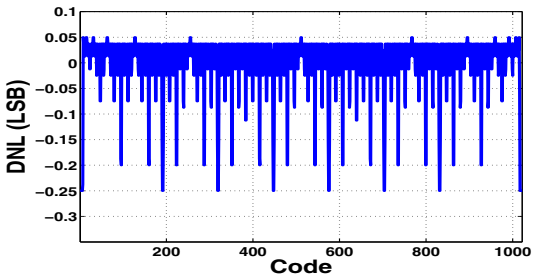
[4] B. C. Kuo, *Automatic Control Systems*. John Wiley & Sons, Inc, 1995.



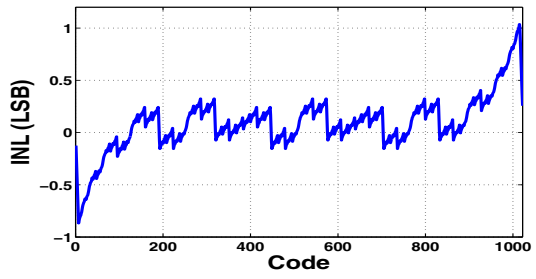
(a)



(b)



(c)



(d)

Fig. 7. Simulation results for a 10-bit ADC: (a) DNL as predicted by our simulator, (b) INL as predicted by our simulator, (c) DNL as predicted by Ultrasim, and (d) INL as predicted by Ultrasim.