EE247
Lecture 12

- Administrative issues
  - Midterm exam Thurs. Oct. 23rd
    - You can *__only__* bring one 8x11 paper with your own written notes (please do not photocopy)
    - No books, class notes or any other kind of handouts/notes, calculators, computers, PDA, cell phones....
    - Midterm includes material covered to end of lecture 14

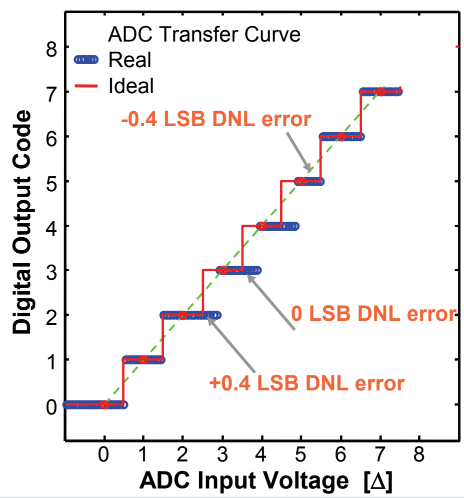
EE247
Lecture 12

- Data converters
  - Static converter error sources (continued)
    - Offset
    - Full-scale error
    - Differential non-linearity (DNL)
    - Integral non-linearity (INL)
  - Measuring DNL & INL
    - Servo-loop
    - Code density testing (histogram testing)
  - Dynamic tests
    - Spectral testing→ Reveals ADC errors associated with dynamic behavior i.e. ADC performance as a function of frequency

ADC Differential Nonlinearity

DNL = deviation of code width from Δ (1LSB)

1. Endpoints connected
2. Ideal characteristics derived eliminating offset & full-scale error
3. DNL measured → code width deviation from 1LSB



ADC Differential Nonlinearity

- Ideal ADC transitions point equally spaced by 1LSB
- For DNL measurement, offset and full-scale error is eliminated
- DNL [k] (a vector) measures the deviation of each code from its ideal width
- Typically, the vector for the entire code is reported
- If only one DNL # is presented that would be the worst case

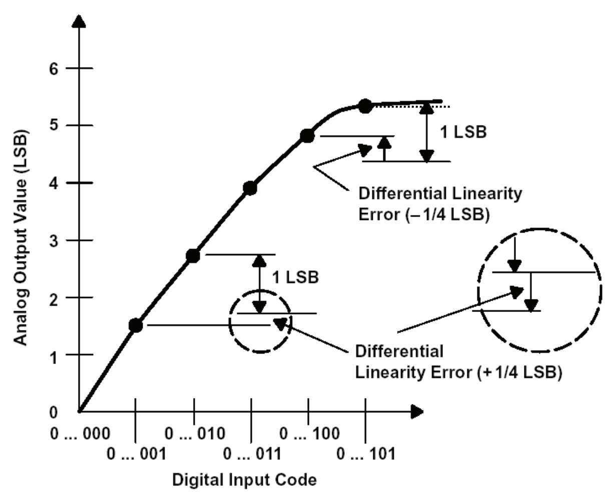
ADC DNL

- DNL=-1 implies missing code
- For an ADC DNL < -1 not possible → undefined
- Can show:

$$\sum^{all\ i} DNL[i] = 0$$

- For a DAC DNL < -1 possible

DAC Differential Nonlinearity



DAC Differential Nonlinearity

- To find DNL for DAC
  - Draw end-point line from 1st point to last
  - Find ideal LSB size for the end-point corrected curve
  - Find segment sizes:

$$segment\,[m]=V[m]-V[m-1]$$

$$DNL[m]=\frac{segment[m]-V[LSB]}{V[LSB]}$$

- Unlike ADC DNL, for a DAC DNL can be <-1LSB

Impact of DNL on Performance

- Same as a somewhat larger quantization error, consequently degrades SQNR
- How much – later in the course...
- The term "DNL noise", usually means "additional quantization noise due to DNL"

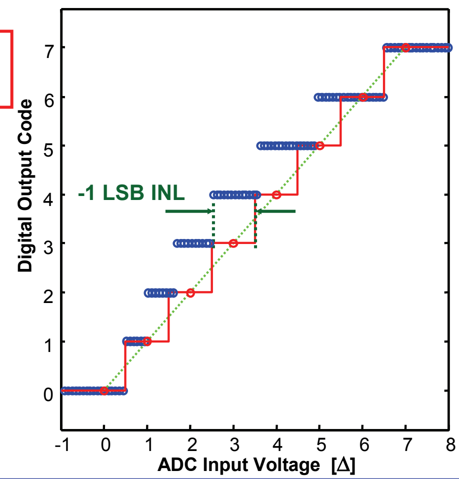
ADC Integral Nonlinearity End-Point

INL = deviation of code transition from its ideal location

1. Endpoints connected

2. Ideal characteristics derived eliminating offset & full-scale error (same as for DNL)

3. INL→ deviation of code transition from ideal is measured



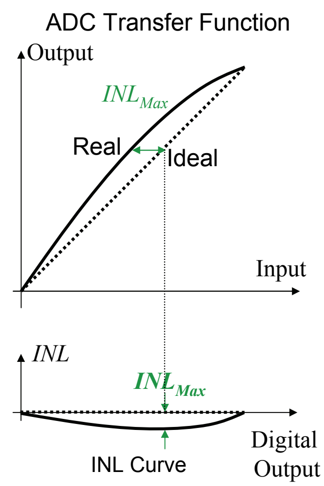
ADC Integral Nonlinearity

INL = deviation of code transition from its ideal location

INL is also a vector INL[k]
If one INL # reported
→Worst case INL

Most common→ End-point:
Straight line through the endpoints is usually used as reference,
i.e. offset and full scale errors are eliminated in INL calculation

Ideal converter steps found for the endpoint line, then INL is measured

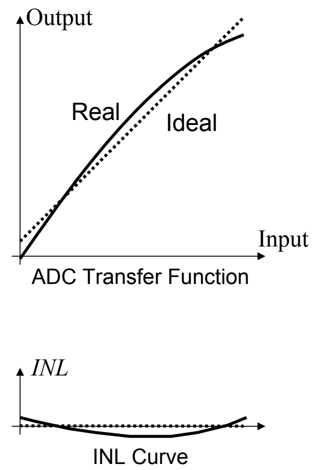


ADC Integral Nonlinearity
Best-Fit

INL = deviation of code transition from its ideal location

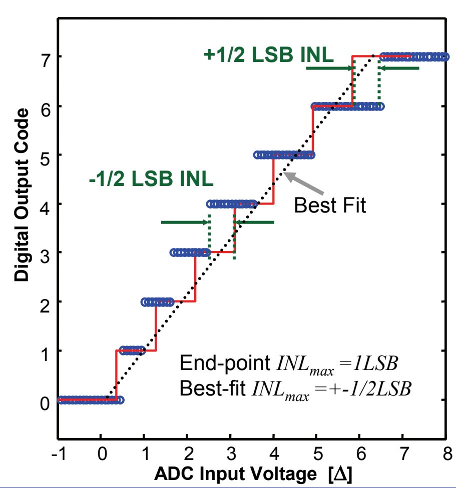
Best-Fit
- A best-fit line (in the least-mean squared sense) fitted to measured data
- Ideal converter steps found then INL measured

Note: Typically INL #s smaller for best-fit compared to end-point



ADC Integral Nonlinearity
Best Fit versus End-Point

- Best-Fit
  - A best-fit line (in the least-mean squared sense)
  - Ideal converter steps is found then INL is measured



ADC Integral Nonlinearity

Can derive INL by:

1-

- Construct uniform staircase between 1st and last transition
- INL for each code:

$$INL[m] = \frac{T[m] - T[ideal]}{W[ideal]}$$

2-

- Can show

$$\boxed{INL[m] = \sum_{i=1}^{m-1} DNL[i]}$$

→ INL is found by computing the cumulative sum of DNL

ADC Differential & Integral Nonlinearity Example

$$INL[m] = \sum_{i=1}^{m-1} DNL[i]$$

Notice:

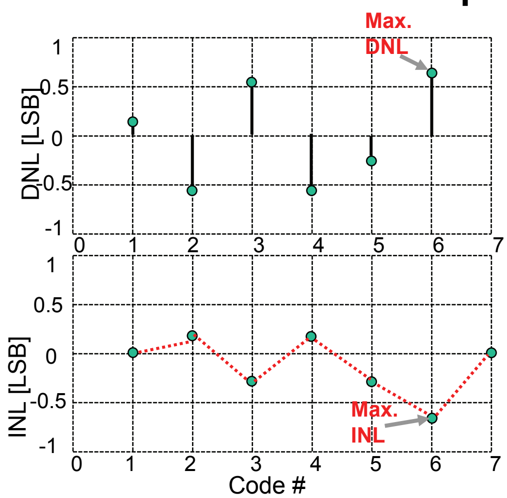
INL[0] → *undefined*

$INL[1]=0$

$INL[2^N-1]=0$

Code #	DNL [LSB]	INL [LSB]
0	-	-
1	0.18	0
2	-0.55	0.18
3	0.55	-0.37
4	-0.55	0.18
5	-0.27	-0.37
6	0.64	-0.64
7	-	0

ADC Differential & Integral Nonlinearity Example



Code #	DNL [LSB]	INL [LSB]
0	-	-
1	*0.18*	*0*
2	*-0.55*	*0.18*
3	*0.55*	*-0.37*
4	*-0.55*	*0.18*
5	*-0.27*	*-0.37*
6	*0.64*	*-0.64*
7	-	*0*

DAC Integral Nonlinearity

Can derive INL by:

- Connect end points
- Find ideal output values
- INL for each code:

$$INL[m] = \frac{V[m] - V[ideal]}{V[LSB]}$$

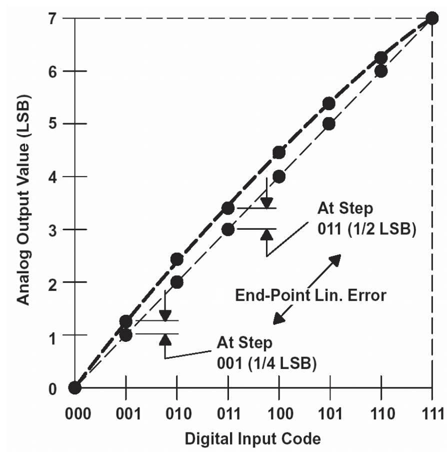
2-

- Can show

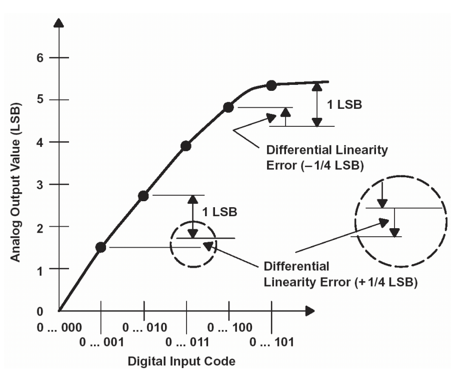
$$\boxed{INL[m] = \sum_{i=1}^{m-1} DNL[i]}$$

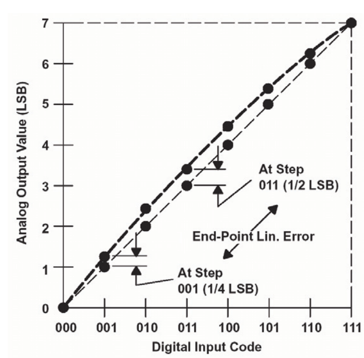
→ INL is found by computing the cumulative sum of DNL

DAC Integral Nonlinearity



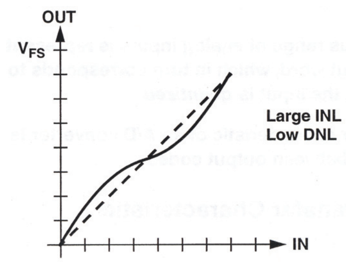
DAC DNL and INL





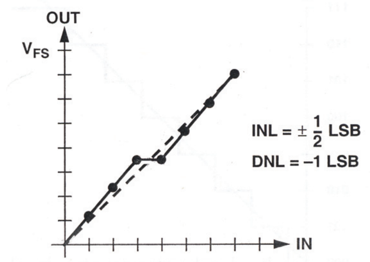
* Ref: "Understanding Data Converters," Texas Instruments Application Report SLAA013, Mixed-Signal Products, 1995.

Example: INL & DNL



Large INL & Small DNL

Smooth variations in transfer curve → Small DNL



Large DNL & Small INL

Abrupt variations in transfer curve → Large DNL

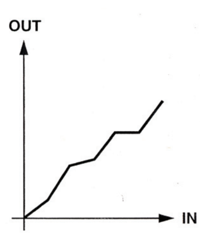
Monotonicity

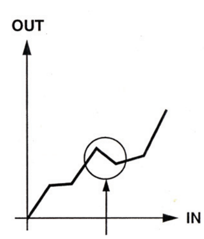
- Monotonicity guaranteed if

  $$|\,\text{INL}\,| \leq 0.5\ \text{LSB}$$

  The best fit straight line is taken as the reference for determining the INL.
- This implies

  $$|\,\text{DNL}\,| \leq 1\ \text{LSB}$$





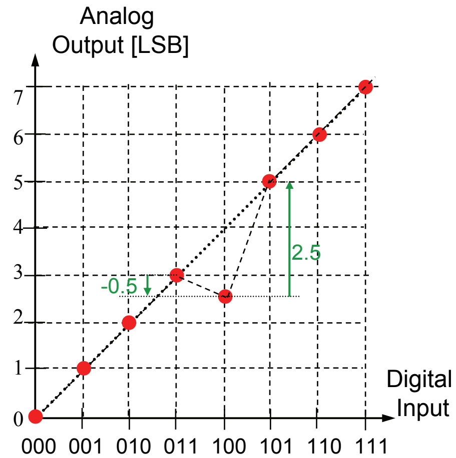
Non-Monotonic DAC

$$DNL[m] = \frac{segment[m] - V[LSB]}{V[LSB]}$$

$$DNL[4] = \frac{segment[4] - V[LSB]}{V[LSB]} = \frac{-0.5 - 1}{1} = -1.5[LSB]$$

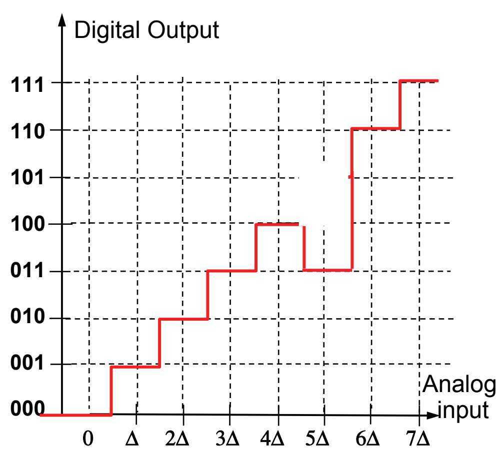
$$DNL[5] == \frac{2.5 - 1}{1} = 1.5[LSB]$$

- **DNL< -1LSB for a DAC → Non-monotonicity**
- When can non-monotonicity cause major problems?



Non-Monotonic ADC

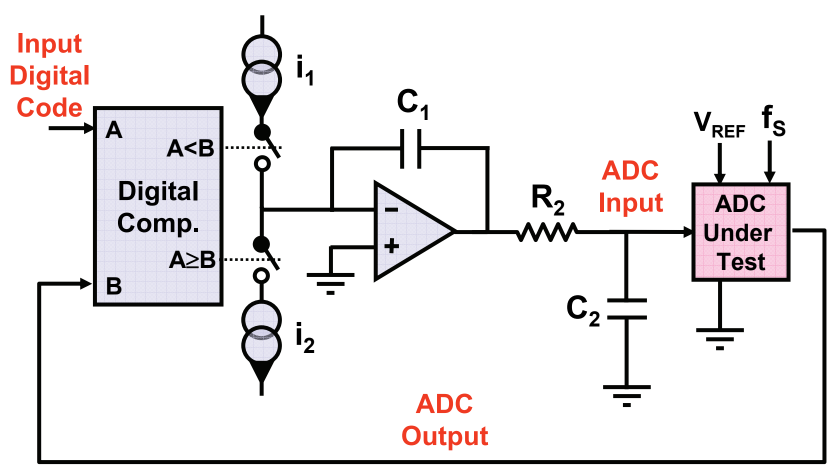
- Code 011 associated with two transition levels !
- For non-monotonic ADC
  →DNL not defined @ non-monotonic steps



How to measure DNL/INL?

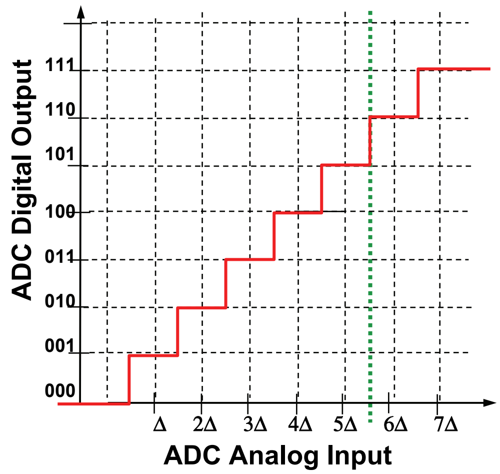
- DAC:
  - Simply apply digital codes and use a good voltmeter to measure corresponding analog output
- ADC
  - Not as simple as DAC→ need to find "decision levels", i.e. input voltages at all code boundaries
    - One way: Adjust voltage source to find exact code trip points "code boundary servo"
    - More versatile: Histogram testing
      →Apply a signal with known amplitude distribution and analyze digital code distribution at ADC output

Code Boundary Servo

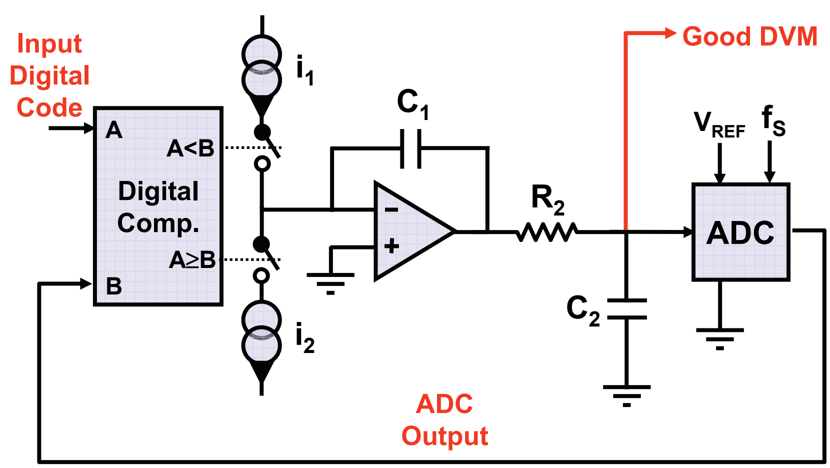


Code Boundary Servo

- i1 and i2 are small, and C1 is large, so the ADC analog input moves a small fraction of an LSB each sampling period
- For a code input of 101, the ADC analog input settles to the code boundary shown



Code Boundary Servo



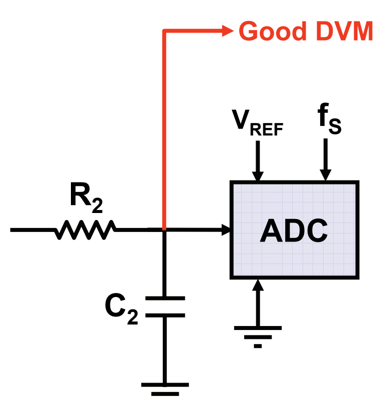
Code Boundary Servo

- A very good digital voltmeter (DVM) measures the analog input voltage corresponding to the desired code boundary
- DVMs have some interesting properties
  - They can have very high resolutions (8½ decimal digit meters are inexpensive)
  - To achieve stable readings, DVMs average voltage measurements over multiple 60Hz ac line cycles to filter out pickup in the measurement loop

Code Boundary Servo

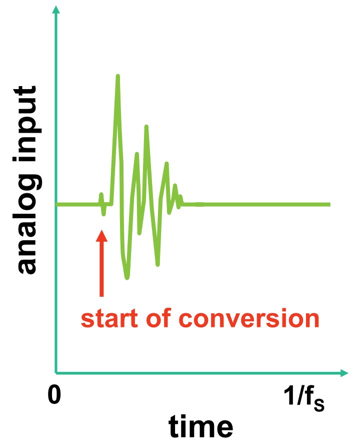
- ADCs of all kinds are notorious for kicking back high-frequency, signal-dependent glitches to their analog inputs

- A magnified view of an analog input glitch follows …



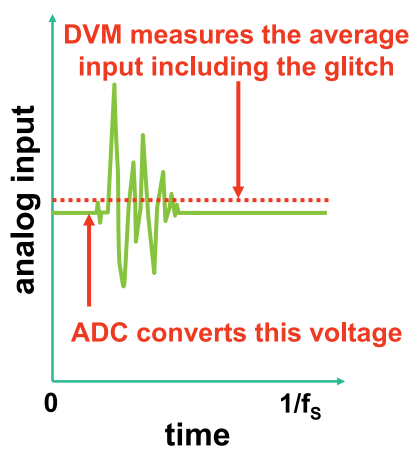
Code Boundary Servo

- Just before the input is sampled and conversion starts, the analog input is pretty quiet
- As the converter begins to quantize the signal, it kicks back charge



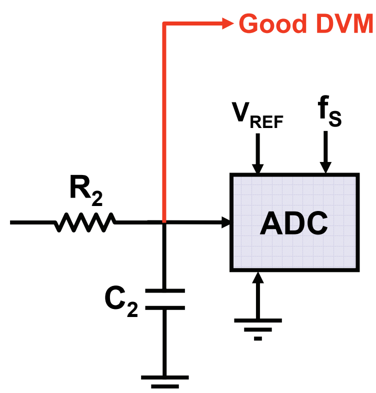
Code Boundary Servo

- The difference between what the ADC measures and what the DVM measures is not ADC INL, it's error in the INL measurement
- How do we control this error?



Code Boundary Servo

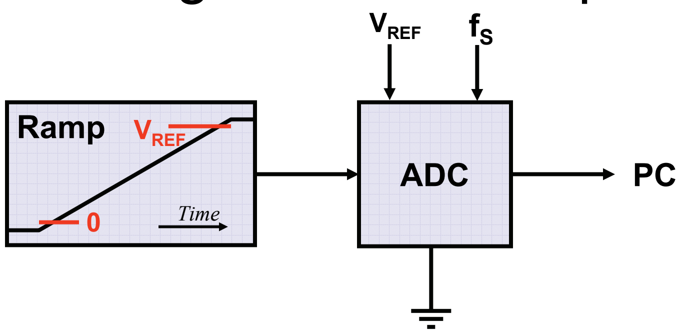
- A large C_2 fixes this
- At the expense of longer measurement time



Histogram Testing

- Code boundary measurements are slow
  - Long testing time
- Histogram testing
  - Quantize input with known pdf (e.g. ramp or sinusoid)
  - Measure output pdf
  - Derive INL and DNL from deviation of measured pdf from expected result

Histogram Test Setup



- Slow (wrt conversion time) linear ramp applied to ADC
- DNL derived directly from total number of occurrences of each code @ the output of the ADC

A/D Histogram Test Using Ramp Signal

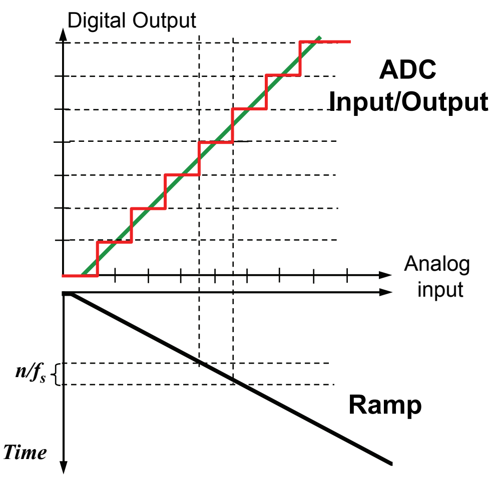
Example:

ADC sampling rate:
f_s =100kHz → T_s=10µsec

1LSB =10mV
For 0.01LSB measurement resolution:
→ n =100 samples/code

→ Ramp duration per code:
=100x10µsec=1msec

→ Ramp slope: 10mV/msec



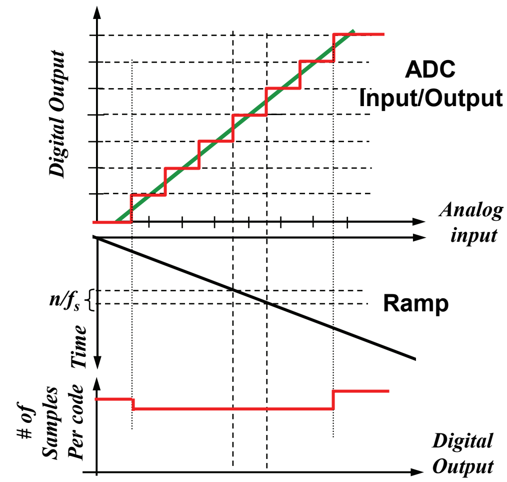
A/D Histogram Test Using Ramp Signal

Example:

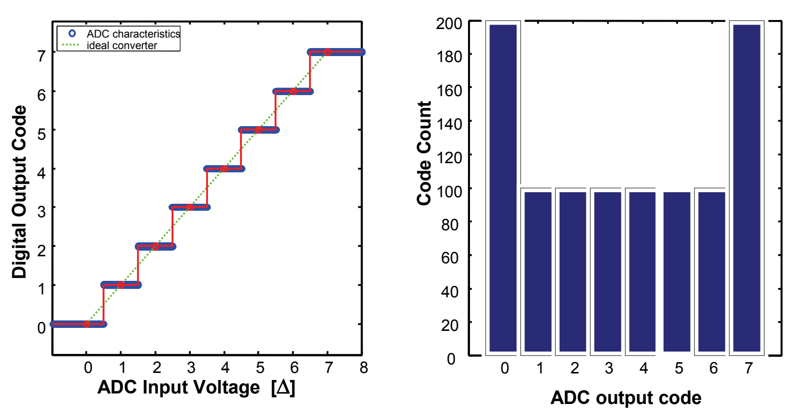
Ramp slope: 10mV/msec
1LSB =10mV
Each ADC code→1msec

f_s =100kHz → T_s=10μsec

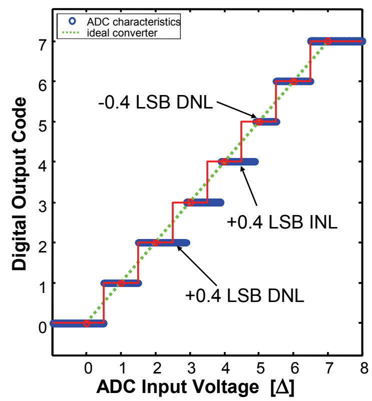
→n =100 samples/code

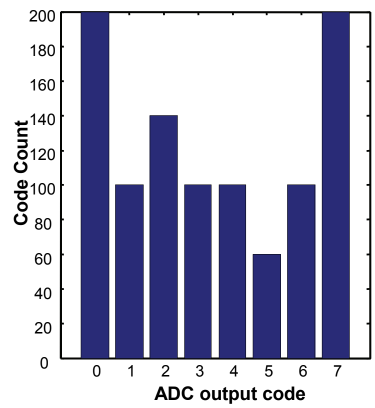


Ramp Histogram
Example: Ideal 3-Bit ADC



Ramp Histogram
Example: Real 3-Bit ADC Including Non-Idealities

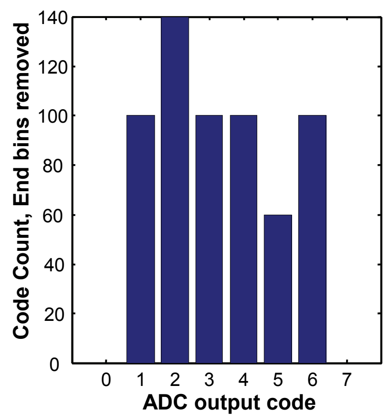




Example: 3 Bit ADC
DNL Extracted from Histogram

1- Remove "Over-range bins" (0 and full-scale)

2- Compute average count/bin (600/6=100 in this case)



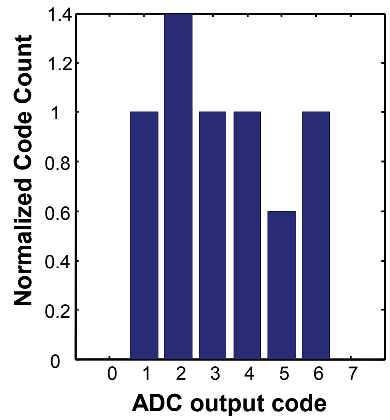
Example: 3 Bit ADC
Process of Extracting from Histogram

3- Normalize:

- Divide histogram by average count/bin

→ ideal bins have exactly the average count, which, after normalization, would be 1

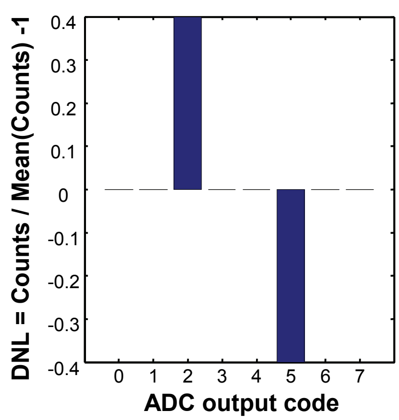
→ Non-ideal bins would have a normalized value greater of smaller than 1



Example: 3 Bit ADC
DNL Extracted from Histogram

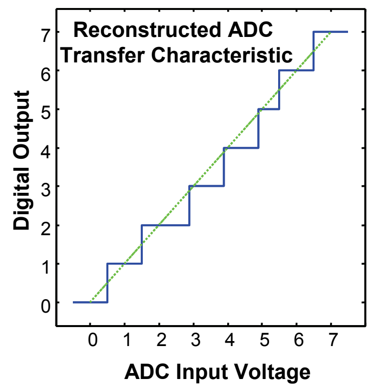
4- Subtract *1* from the normalized code count

5- Result → DNL (+-0.4Lsb in this case)

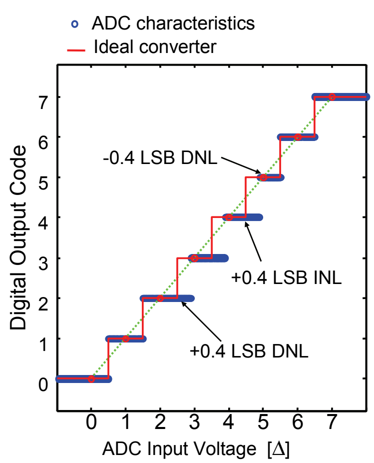


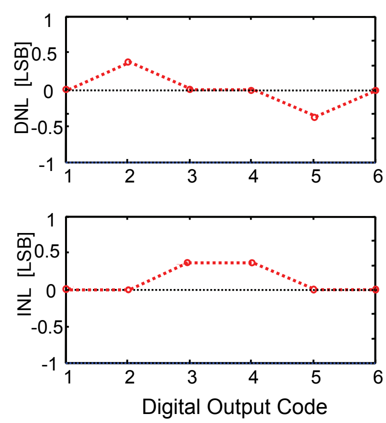
Example: 3-Bit ADC
Static Characteristics Extracted from Histogram

- DNL histogram → used to reconstruct the exact converter characteristic (having measured only the histogram)
- Width of all codes derived from measured DNL (Code=DNL + 1LSB)
- INL→(deviation from a straight line through the end points)- is found



Example: 3 Bit ADC
DNL & INL Extracted from Histogram





Measuring DNL

- Ramp speed is adjusted to provide large number of output/code - e.g. an average of 100 outputs of each ADC code (for 1/100 LSB resolution)

- Ramp test can be quite slow for high resolution ADCs
- Example:
  16bit ADC & 100conversions/code @100kHz sampling rate

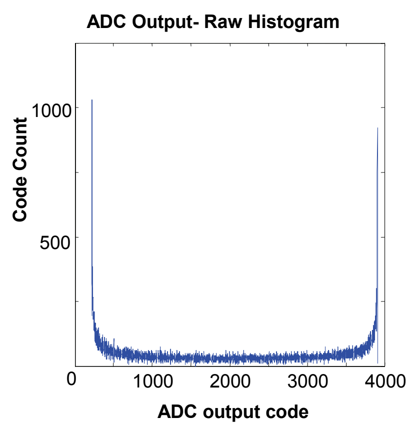
$$\frac{\mathbf{(2^{16}\text{ or } 65{,}536 \text{ codes})(100 \text{ conversions/code})}}{\mathbf{100{,}000 \text{ conversions/sec}}} = \mathbf{65.6 \text{ sec}}$$

ADC Histogram Testing Sinusoidal Inputs

- Ramp signal generators linear to only 8 to10bits
  → Need to find input signal with better purity

- Solution:
  →Use sinusoidal test signal (may need to filter out harmonics)

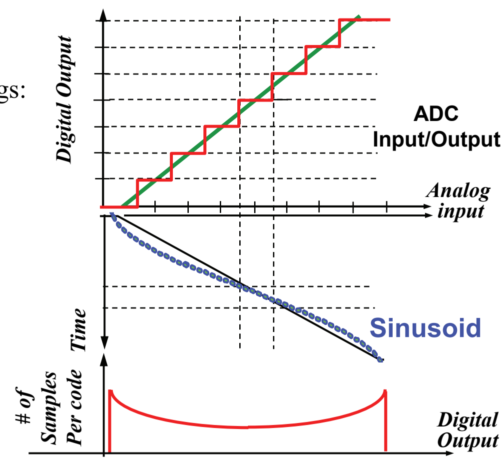
- Problem: Ideal ADC histogram not flat but has “bath-tub shape”



ADC Histogram Test Using Sinusoidal Signals

At sinusoid midpoint crossings:
dv/dt → max.
→ least # of samples

At sinusoid amplitude peaks:
dv/dt → min.
→ highest # of samples



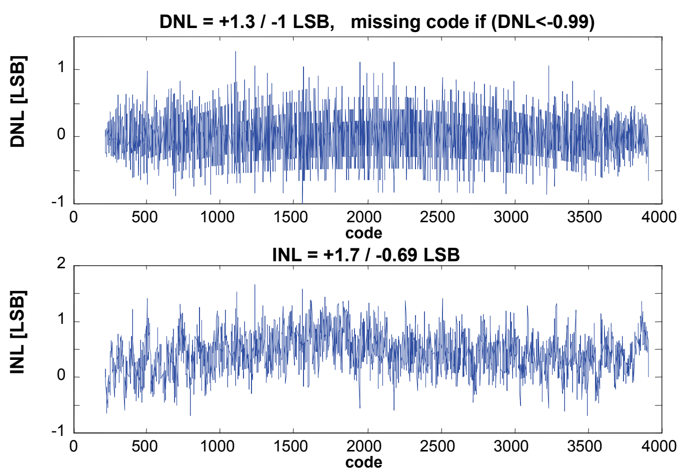
Histogram Testing
Correction for Sinusoidal PDF

- References:
  - [1] M. V. Bossche, J. Schoukens, and J. Renneboog, "Dynamic Testing and Diagnostics of A/D Converters," IEEE Transactions on Circuits and Systems, vol. CAS-33, no. 8, Aug. 1986.
  - [2] IEEE Standard 1057
- Is it necessary to know the exact amplitude and offset of sinusoidal input? No!

DNL/INL Extraction Matlab Program

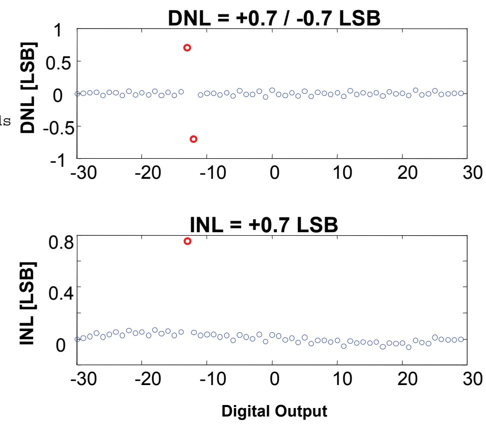
```
function [dnl,inl] = dnl_inl_sin(y);
%DNL_INL_SIN
% dnl and inl ADC output
% input y contains the ADC output
% vector obtained from quantizing a
% sinusoid

% Boris Murmann, Aug 2002
% Bernhard Boser, Sept 2002

% histogram boundaries
minbin=min(y);
maxbin=max(y);

% histogram
h = hist(y, minbin:maxbin);

% cumulative histogram
ch = cumsum(h);

% transition levels found by:
T = -cos(pi*ch/sum(h));

% linearized histogram
hlin = T(2:end) - T(1:end-1);

% truncate at least first and last
% bin, more if input did not clip ADC
trunc=2;
hlin_trunc = hlin(1+trunc:end-trunc);

% calculate lsb size and dnl
lsb= sum(hlin_trunc) / (length(hlin_trunc));
dnl= [0 hlin_trunc/lsb-1];
misscodes = length(find(dnl<-0.99));

% calculate inl
inl= cumsum(dnl);
```


Example: Test Results for DNL & INL Using Sinusoidal Histogram



Example: Matlab ADC Model DNL/INL Code Test

```
% converter model
B = 6;                % bits
range = 2^(B-1) - 1;
% thresholds (ideal converter)
th = -range:range; % ideal thresholds
th(20) = th(20)+0.7; % error

fs = 1e6;
fx = 494e3 + pi;      % try fs/10!
C  = round(100 * 2^B / (fs / fx));

t = 0:1/fs:C/fx;
x = (range+1) * sin(2*pi*fx.*t);
y = adc(x, th) - 2^(B-1);

hist(y, min(y):max(y));

dnl_inl_sin(y);
```



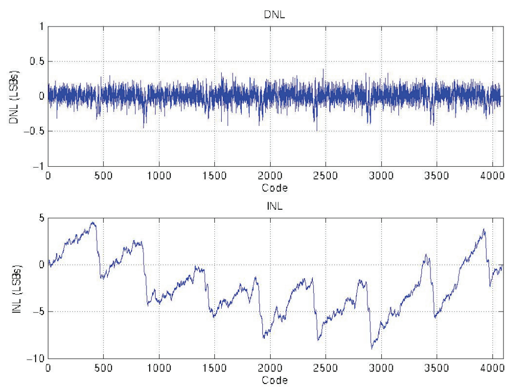
Histogram Testing Limitations

- The histogram (as any ADC test, of course) characterizes one particular converter. Test many devices to get valid statistics.
- Histogram testing assumes monotonicity
  E.g. "code flips" will not be detected.
- Dynamic sparkle codes produce only minor DNL/INL errors
  E.g. 123, 123, …, 123, <u>0</u>, 124, 124, … → look at ADC output to detect
- Noise not detected & averaged out
  E.g. 9, 9, 9, <u>10</u>, 9, 9, 9, 10, <u>9</u>, 10, 10, 10, …

Ref: B. Ginetti and P. Jespers, "Reliability of Code Density Test for High Resolution ADCs," Electron. Lett., vol. 27, pp. 2231-3, Nov. 1991.

Example: Hiding Problems in the Noise

- INL → 5 missing codes
- DNL "smeared out" by noise!
- Always look at both DNL/INL
- INL usually does not lie...

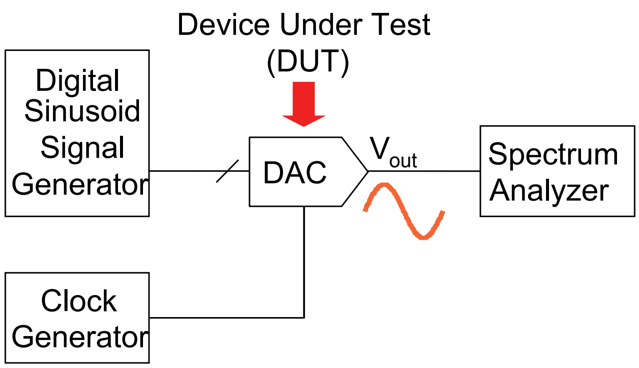


[Source: David Robertson, Analog Devices]

Why Additional Tests/Metrics?

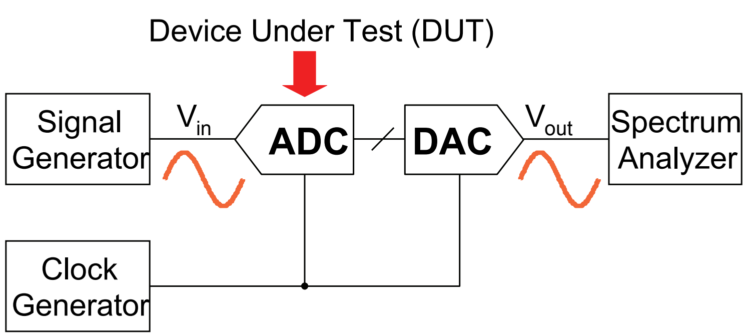
- Static testing does not tell the full story
  – E.g. no info about "noise“ or high frequency effects
- Frequency dependence (f_s and f_{in}) ?
  – In principle we can vary f_s and f_{in} when performing histogram tests
  – Result of such sweeps is usually not very useful
  – Hard to separate error sources, ambiguity
  – Typically we use $f_s=f_{sNOM}$ and $f_{in} << f_s/2$ for histogram tests
- For additional info→ Spectral testing

DAC Spectural Test or Simulation



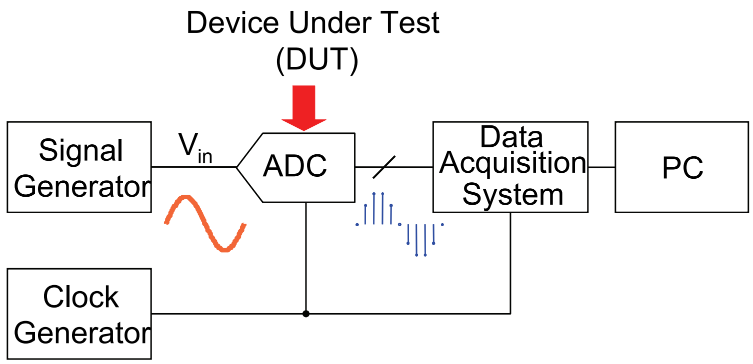
- Input sinusoid → Need to have significantly better purity compared to DAC linearity
- Spectrum analyzer need to have better linearity than DUT

Direct ADC Spectral Test via DAC

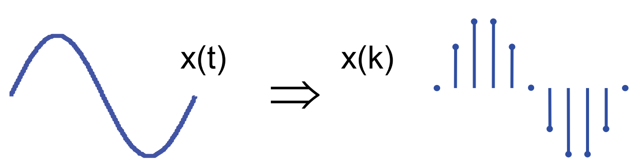


- Need DAC with much better performance compared to ADC under test
- Beware of DAC output sinx/x frequency shaping
- Good way to "get started"...

ADC Spectral Test via Data Acquisition Sytem



Analyzing ADC Outputs via Discrete Fourier Transform (DFT)



- Sinusoidal waveform has all its power at one single frequency
- An ideal, infinite resolution ADC would preserve ideal, single tone spectrum
- DFT used as a vehicle to reveal ADC deviations from ideality

Discrete Fourier Transform

The DFT of a block of N time samples

$\{x(k)\} = \{x(0), x(1), x(2),\ldots,x(N-1)\}$

yields a set of N frequency bins

$\{A_m\} = \{A_0,A_1,A_2,\ldots,A_{N-1}\}$

where:

$$A_m = \sum_{n=0}^{N-1} x_n W_N^{mn} \qquad m = 0,1,2,\ldots,N-1$$

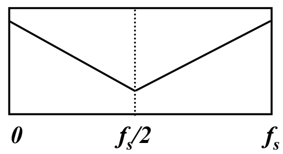
$$W_N \equiv e^{-j2\pi/N}$$

Discrete Fourier Transform (DFT) Properties

- DFT of N samples spaced $T_s=1/f_s$ seconds:
  - N frequency bins from DC to f_s
  - Bin m represents frequencies at $m * f_s/N$ $[Hz]$
- DFT frequency resolution:
  - Proportional to f_s/N in [Hz/bin]
- DFT with $N = 2^k$ (k is an integer) can be found using a computationally more efficient algorithm named:
  - FFT → Fast Fourier Transform

DFT Magnitude Plots

- Because magnitudes of DFT bins (A_m) are symmetric around $f_S/2$, it is redundant to plot $|A_m|$'s for $m > N/2$



- Usually magnitudes are plotted on a log scale normalized so that a full scale sinusoidal waveform with *rms* value a_{FS} yields a peak bin of *0dBFS*:

$$|A_m| \ [dBFS] = 20 \log_{10} \frac{|A_m|}{a_{FS} \cdot N/2}$$

Matlab Example Normalized DFT

```
fs  = 1e6;
fx  = 50e3;
Afs = 1;
N   = 100;

% time vector
t = linspace(0, (N-1)/fs, N);
% input signal
y = Afs * cos(2*pi*fx*t);
% spectrum
s = 20 * log10(abs(dft(y)/N/Afs*2));
% drop redundant half
s = s(1:N/2);
% frequency vector (normalized to fs)
f = (0:length(s)-1) / N;
```