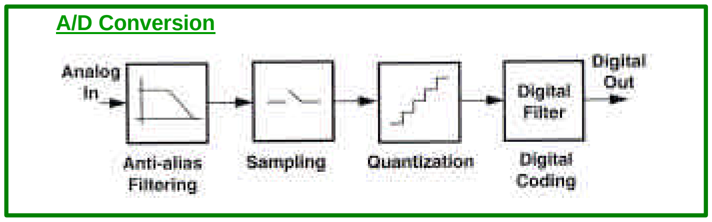
EE247
Lecture 12

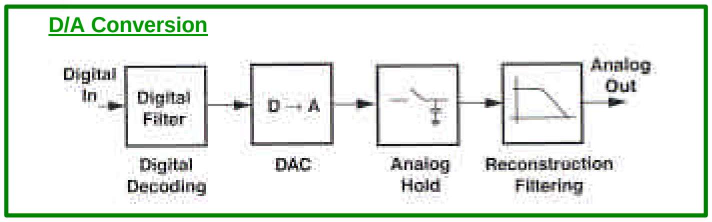
- Administrative issues
  - Midterm exam Oct. 19th.
    - o You can *only* bring one 8x11 paper with notes
    - o No books, class handouts, calculators, computers, cell phones....
  - Final exam date in process of changing- feedback so far from students the only conflicting other final is EE142- if you have any other finals last chance to announce

EE247
Lecture 12

- Data Converters
  - Summary last lecture
  - ADC & DAC testing
    - DNL & INL
      - Code boundry servo test
      - Histogram testing
    - Spectral testing

A/D & D/A Conversion



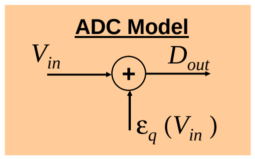


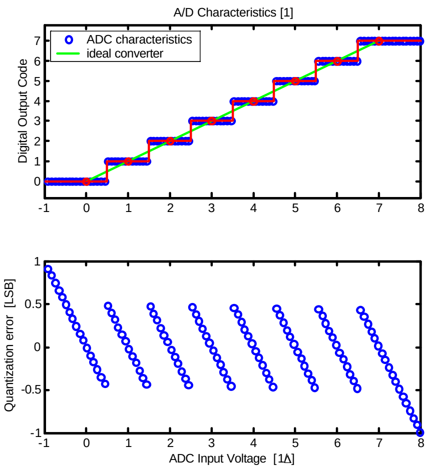
Classification

- $f_s > 2f_{max}$ Nyquist Sampling
  - "Nyquist Converters"
  - Actually always slightly oversampled
- $f_s >> 2f_{max}$ Oversampling
  - "Oversampled Converters"
  - Anti-alias filtering is often trivial
  - Oversampling is also used to reduce quantization noise, see later in the course...
- $f_s < 2f_{max}$ Undersampling (Subsampling)

Ideal ADC ("Quantizer")

- Quantization step Δ (= 1 LSB)
- E.g. N = 3 Bits
- Full-scale input range: -0.5Δ … (2^N-0.5)Δ
- Quantization error: bounded by –Δ/2 … +Δ/2 for inputs within full-scale range





ADC Signal-to-Quantization Noise Ratio

- If certain conditions are met, quantization error can be viewed as being "random", and is often referred to as “noise”
- In this case, we can define a peak “signal-to-quantization noise ratio”, SQNR, for sinusoidal inputs:

$$SQNR = \frac{\frac{1}{2}\left(\frac{2^N \Delta}{2}\right)^2}{\frac{\Delta^2}{12}} = 1.5 \times 2^{2N}$$

$$\boxed{= 6.02N + 1.76 \text{ dB}}$$

e.g.	N	SQNR
	8	50 dB
	12	74 dB
	16	98 dB
	20	122 dB

- Actual converters do not quite achieve this performance due to other errors, including
  - Electronic noise
  - Deviations from the ideal quantization levels

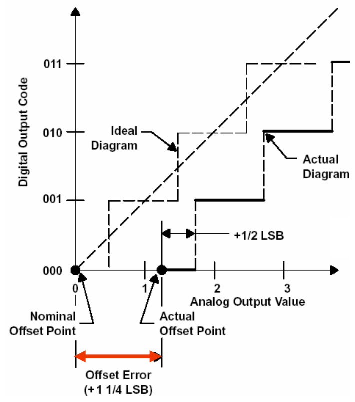
Static Converter Errors

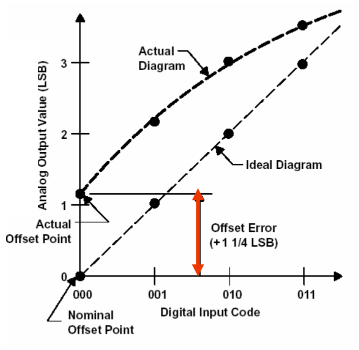
Deviations of characteristic from ideal

- Offset
- Full-scale error
- Differential nonlinearity, DNL
- Integral nonlinearity, INL

Offset Error

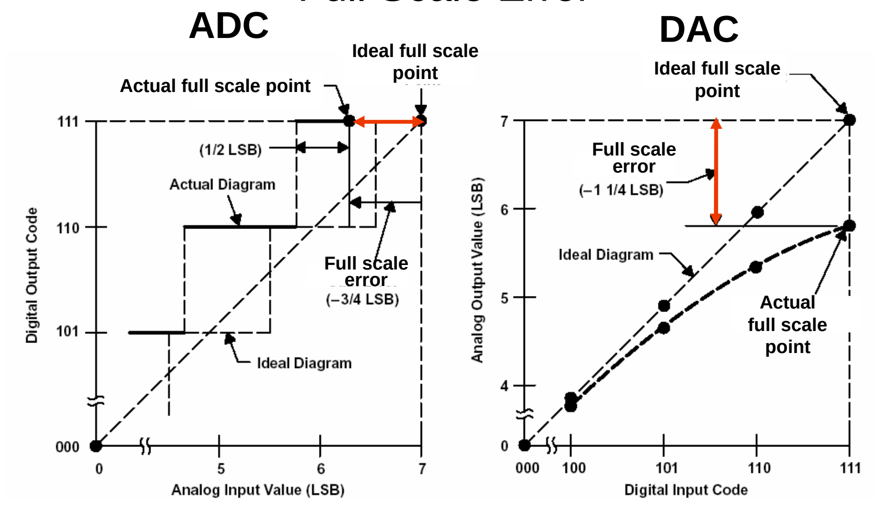
ADC **DAC**





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

Full Scale Error

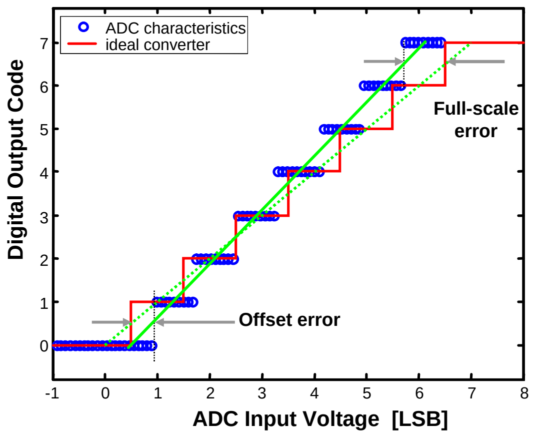


Offset and Full Scale Errors

- Alternative Specification in % Full Scale = 100% * (LSB value)/ 2^N
- Gain error can be extracted from offset & full-scale error
- Non-trivial to build a converter with extremely good offset/full-scale specs
- Typically offset/full-scale is most easily compensated by the digital pre/post-processor
- More interesting: Linearity →DNL, INL

Offset and Full-Scale Error

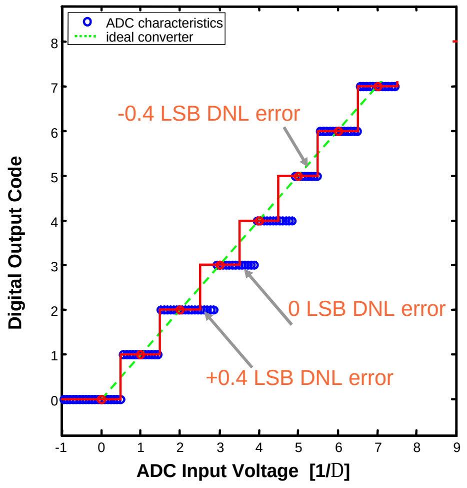
Note:
→ For further measurements (DNL, INL) connecting the endpoints & deriving ideal codes based on the non-ideal endpoints elliminates offset and full-scale error



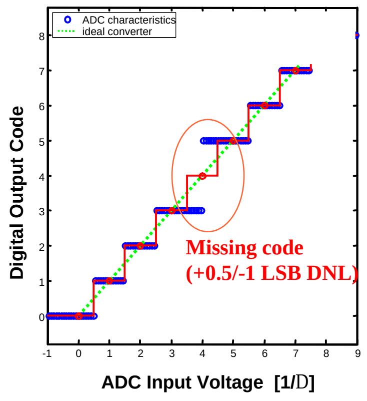
ADC Differential Nonlinearity

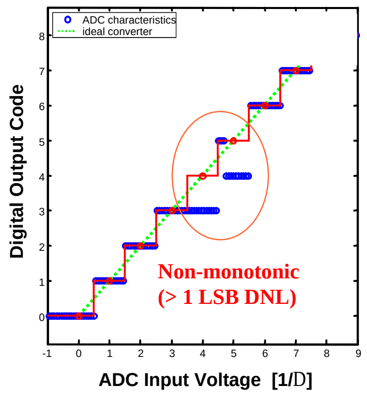
DNL = deviation of code width from Δ (1LSB)

→ Endpoints connected
→ Ideal characteristics derived
→ DNL measured

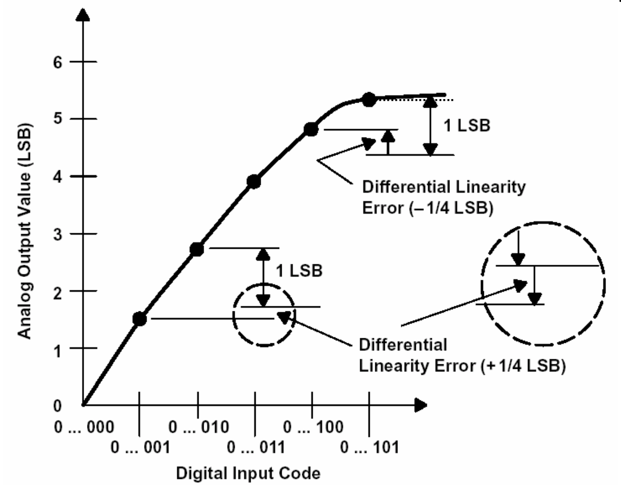


ADC Differential Nonlinearity Examples





DAC Differential Nonlinearity


Analog Output Value (LSB)
1 LSB
Differential Linearity Error (– 1/4 LSB)
1 LSB
Differential Linearity Error (+ 1/4 LSB)
0 ... 000
0 ... 001
0 ... 010
0 ... 011
0 ... 100
0 ... 101
Digital Input Code

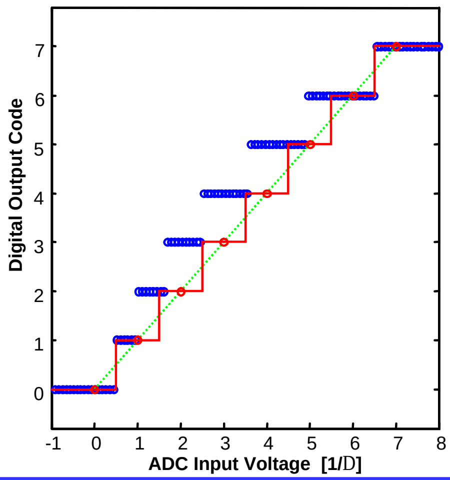
Impact of DNL on Performance

- Same as a somewhat larger quantization error, consequently degrades SQNR
- How much – later in the course...
- People sometimes speak of "DNL noise", i.e. "additional quantization noise due to DNL"

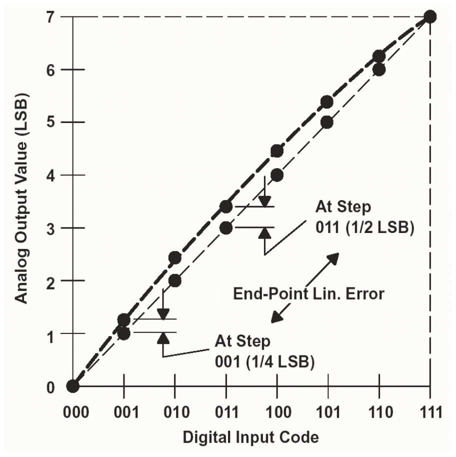
ADC Integral Nonlinearity

INL = deviation of code transition from its ideal location

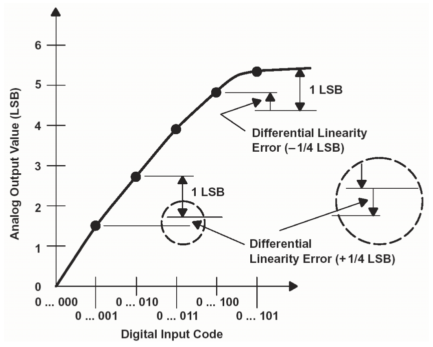
- A straight line through the endpoints is usually used as reference,
  i.e. offset and full scale errors are ignored in INL calculation
- Ideal converter steps is found for the endpoint line, then INL is measured
- Note that INL errors can be much larger than DNL errors and vice-versa

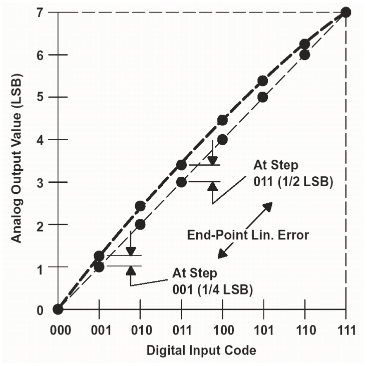


DAC Integral Nonlinearity



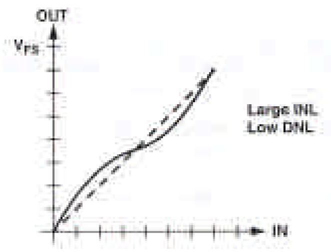
DAC DNL and INL



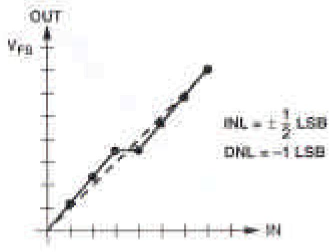


* Ref: "Understanding Data Converters," Texas Instruments Application Report SLAA013, Mixed-Signal Products, 1995.

Example: INL & DNL



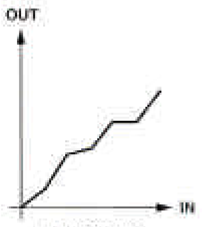
Large INL & Small DNL



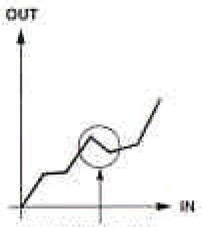
Large DNL & Small INL

Monotonicity

- Monotonicity guaranteed if
  | INL | = 0.5 LSB
  The best fit straight line is taken as the reference for determining the INL.
- This implies
  | DNL | = 1 LSB
- Note: these conditions are *sufficient* but not *necessary* for monotonicity



MONOTONIC



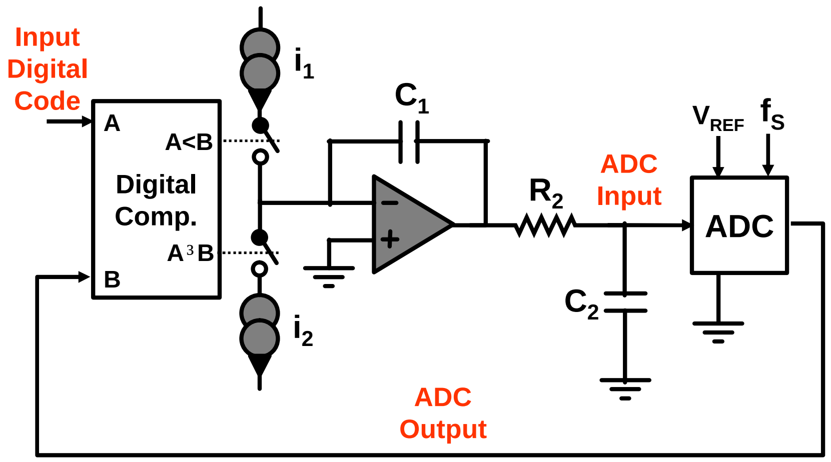
NON-MONOTONIC

How to measure DNL/INL?

- DAC:
  - "trivial", apply codes and use a good voltmeter to measure output
- ADC
  - 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

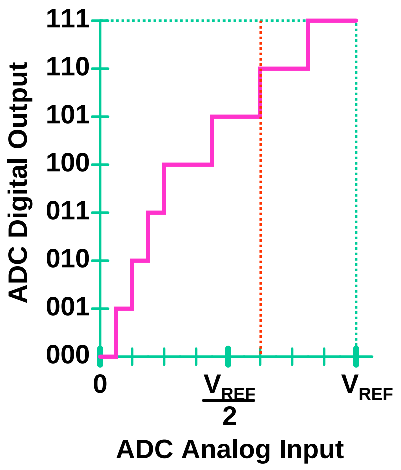
      →Apply a signal with known distibution and analyze digital code distribution at ADC output

Code Boundary Servo

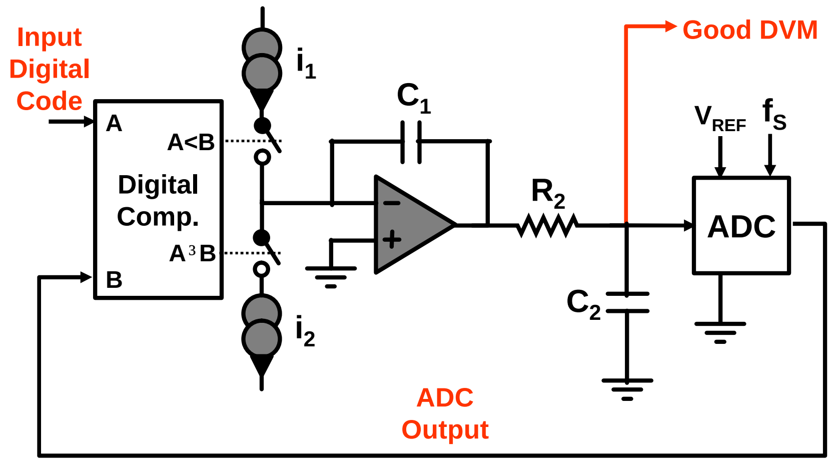


Code Boundary Servo

- i_1 and i_2 are small, and C_1 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

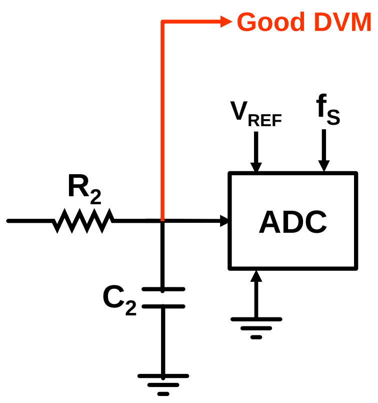


Code Boundary Servo

- 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

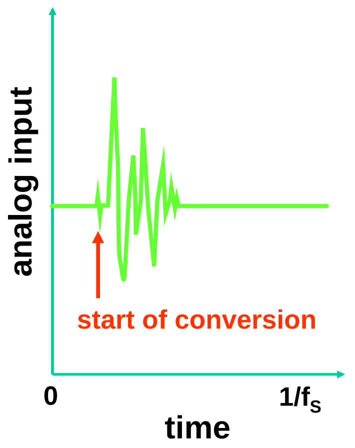
Code Boundary Servo

- 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 …



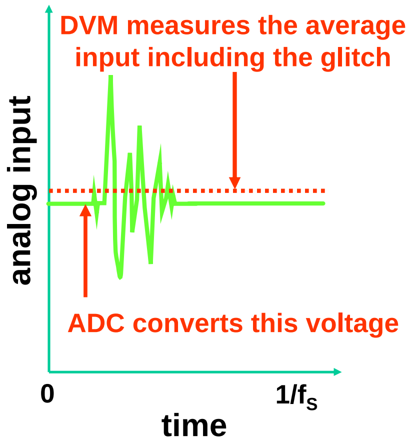
Code Boundary Servo

- 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



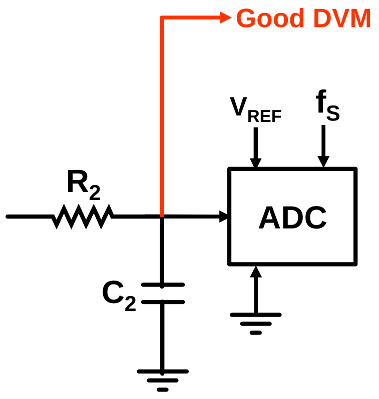
Code Boundary Servo

- 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?


DVM measures the average input including the glitch
analog input
ADC converts this voltage
0
$1/f_S$
time

Code Boundary Servo

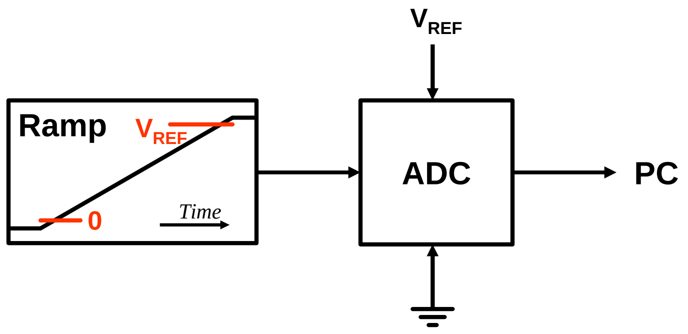
- A large C_2 fixes this
- At the expense of longer measurement time



Histogram Testing

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

Histogram Test Setup



- DNL follows directly from total number of occurrences of each code

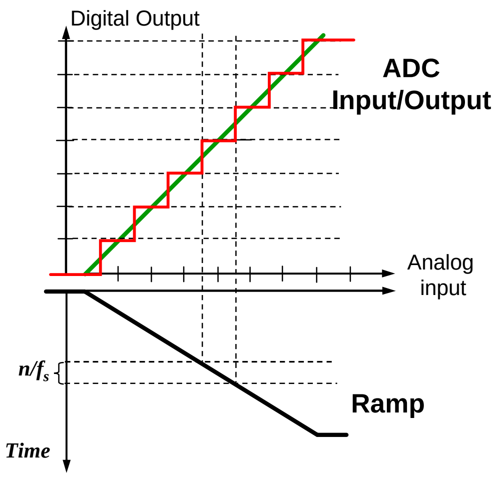
A/D Histogram Test Using Ramp Signal

Example:

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

f_s =100kHz → T_s=10µsec

→ n =100 samples/code



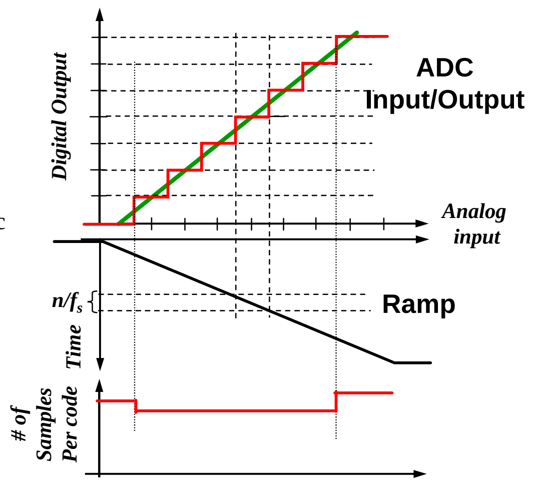
A/D Histogram Test Using Ramp Signal

Example:

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

f_s =100kHz → T_s=10µsec

→ n =100 samples/code



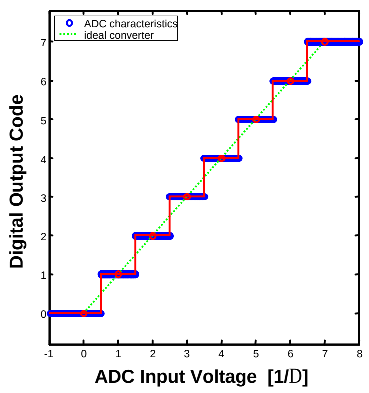
Measuring DNL Error

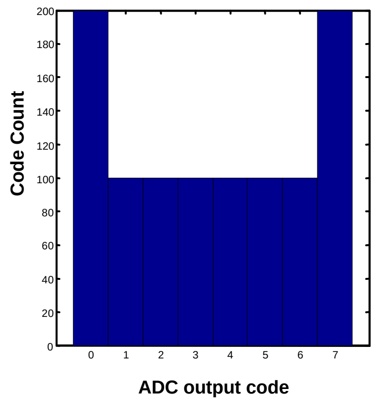
- Ramp speed is adjusted to provide e.g. an average of 100 outputs of each ADC code
  (for 1/100 LSB resolution)

- Ramps can be quite slow for high resolution ADCs
- Example:
  16bit ADC & 100conversion/code @100kHz

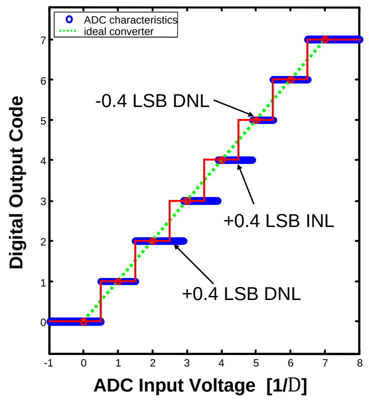
$$\frac{\textbf{(65,536 codes)(100 conversions/code)}}{\textbf{100,000 conversions/sec}} = \textbf{65.6 sec}$$

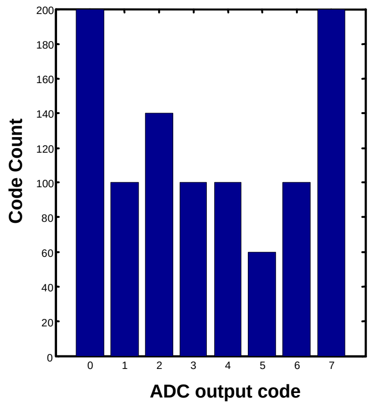
Ramp Histogram
Ideal 3 Bit ADC





Ramp Histogram
Example 3 Bit ADC

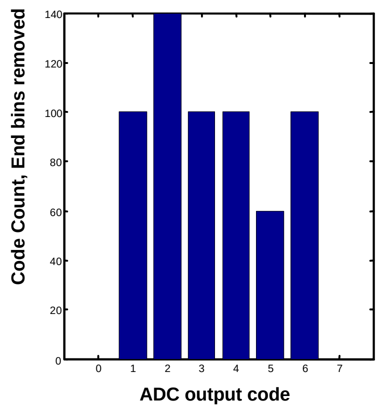




Example 3 Bit ADC DNL Extracted from Histogram

Remove “over-range bins” (0 and full-scale)

Compute average count/bin

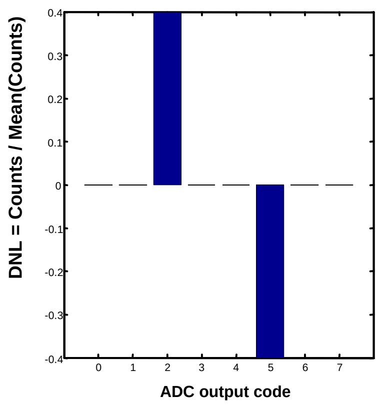


Example 3 Bit ADC DNL Extracted from Histogram

Scale:

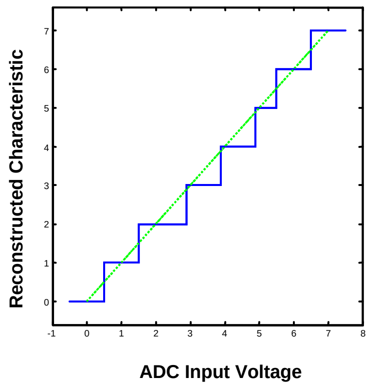
1. divide by average count
2. subtract 1 (ideal bins have exactly the average count, which, after normalization, is 1)

Result is DNL

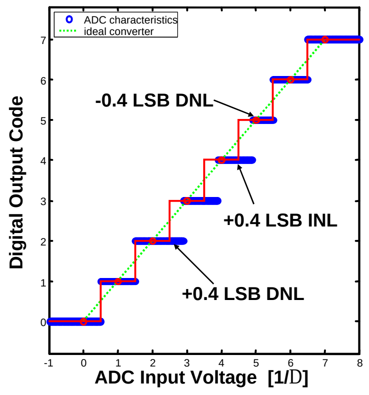


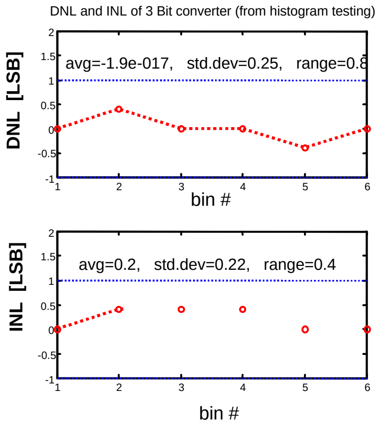
Example 3 Bit ADC
INL Extracted from Histogram

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



Example 3 Bit ADC
DNL & INL Extracted from Histogram

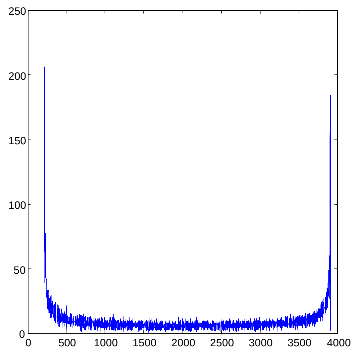




ADC Histogram Testing
Sinusoidal Inputs

- Precise ramps not readily available
- Solution:
  →use sinusoidal test signal
- Problem: ideal histogram is not flat but has "bath-tub shape"

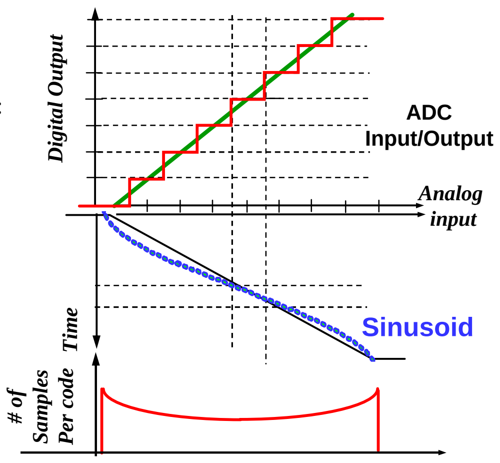
ADC Output- Raw Histogram



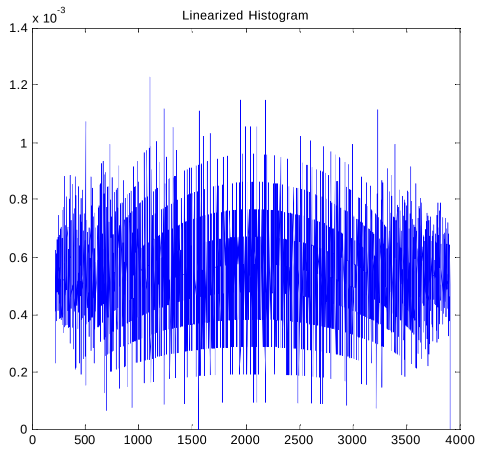
A/D Histogram Test Using Sinusoidal Signals

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

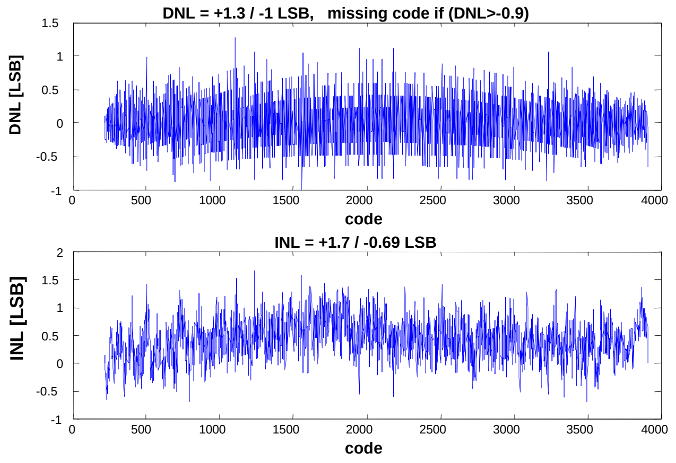
At sinusoid amplitude peaks:
dv/dt→ min.
→ highest # of samples



After Correction for Sinusoidal pdf



Resulting DNL and INL



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 sine input? No!

DNL/INL Code

```
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
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<-0.9));

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

DNL/INL Code Test

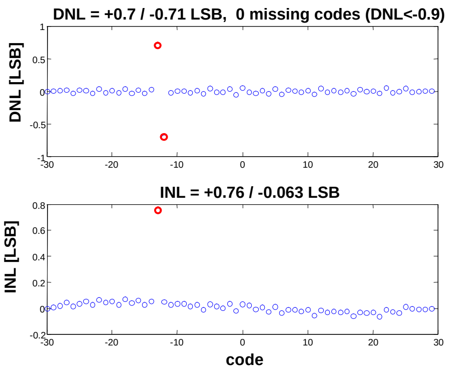
```
% converter model
B = 6;                % bits
range = 2^(B-1) - 1;
% thresholds (ideal converter)
th = -range:range;   % ideal thresholds
th(20) = th(20)+0.7; % error

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);
```



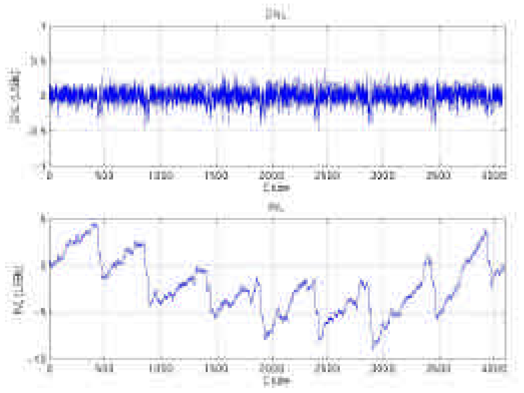
Limitations of Histogram Testing

- 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 or improves DNL.
  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.

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...

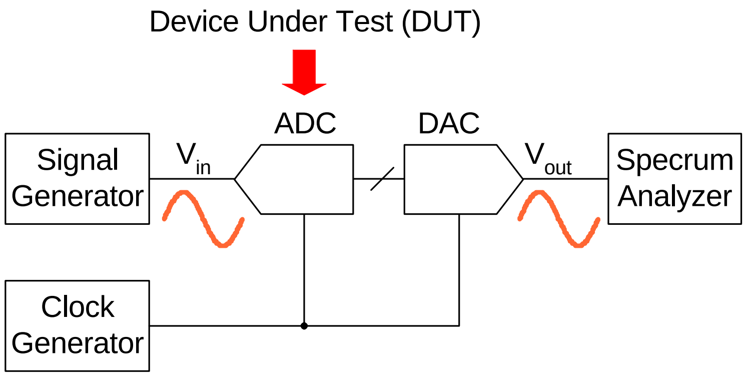


[Source: David Robertson, Analog Devices]

Why Additional Tests/Metrics?

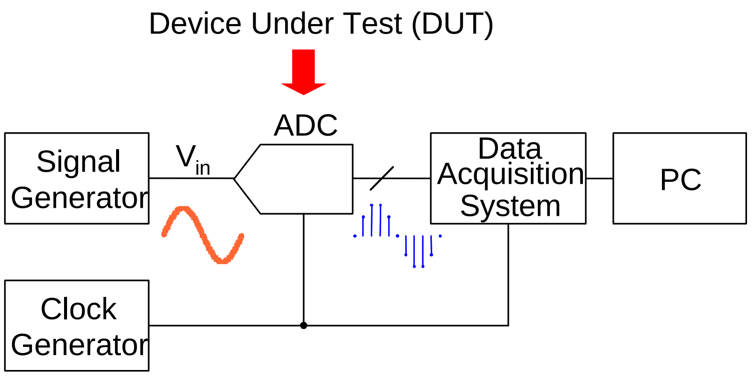
- Static testing does not tell the full story
  – E.g. no info about "noise"
- Frequency dependence (f_s and f_{in}) ?
  – In principle we can vary f_s and f_{in} when performing histogram tests
  – Result of such sweeps is usually not very useful
  – Hard to separate error sources, ambiguity
  – Typically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

Direct ADC-DAC Test

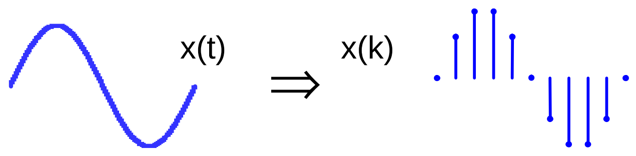


- Need DAC with much better performance compared to ADC under test
- Actually a good way to "get started"...

DFT Test



Analyzing ADC outputs via DFT



- An ideal, infinite resolution ADC would preserve ideal, single tone spectrum
- Deviations reveal ADC non-idealities

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:

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

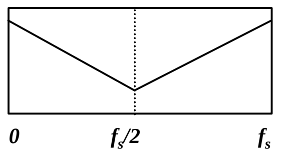
$$\mathbf{W_N \equiv e^{j2\pi/N}}$$

DFT Properties

- DFT of N samples spaced $T=1/f_s$ seconds:
  - N frequency bins
  - Bin m represents frequencies at m * f_S/N [Hz]
- DFT frequency resolution:
  - Proportional to 1/(NT)   in [Hz/bin]

DFT Magnitude Plots

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



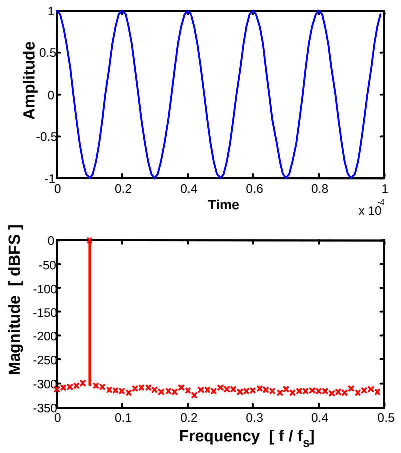
- Usually magnitudes are plotted on a log scale normalized so that a full scale sinewave of rms value a_{FS} yields a peak bin of 0dBFS:

$$|A_m| \text{ (dBFS)} = 20 \log_{10} \frac{|A_m|}{a_{FS}\, N/2}$$

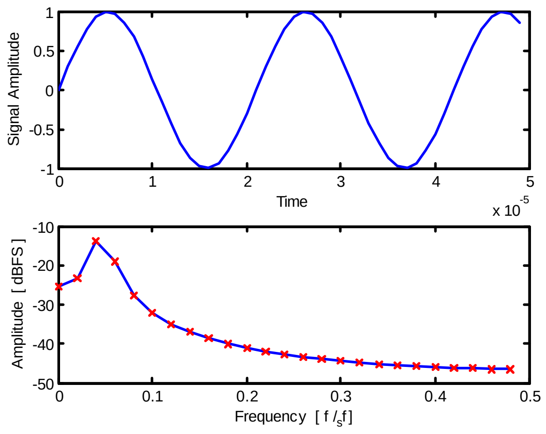
Normalized DFT

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

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



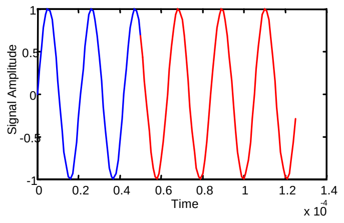
"Another" Example …

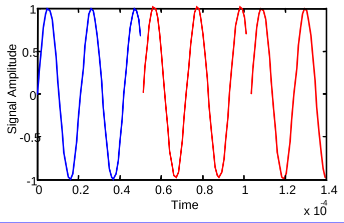


This does not look like the spectrum of a sinusoid …

DFT Periodicity

- The DFT implicitly assumes that time sample blocks repeat every N samples
- With a non-integral number of periods periods within our observation window, the input yields a huge amplitude/phase discontinuity at the block boundary
- This energy spreads into all frequency bins as “spectral leakage”
- Spectral leakage can be eliminated by either
  - An integral number of sinusoids in each block
  - Windowing





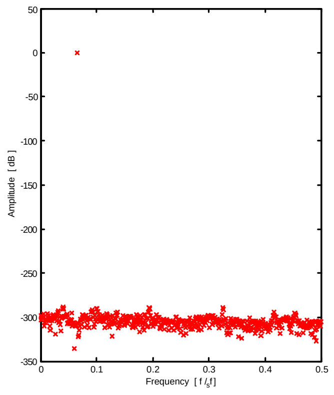
Integral Number of Periods

```
fs = 1e6;

% number of full cycles in test
cycles = 67;

% power of 2 speeds up analysis
% but make N/cycles non-integer!
N = 2^10;

% signal frequency
fx = fs*cycles/N
```


Amplitude [dB]
Frequency [f / fs]