ACT

ADC Characterization Toolkit

Federico Asara
Juan David Gonzalez Cobas

August 15, 2011

In this document I will provide a brief introduction to ADCs and their performance parameters, along with the procedures to calculate them.

Then I will present you ACT, the ADC Characterization Toolkit, a Python-powered application to evaluate the performances of ADCs. The goal of this tool is to work with three different kind of signals: a single sinusoidal wave, to evaluate static and dynamic parameters; two sinusoids with very close frequencies, to evaluate the intermodal distortion; sinusoidal wave with frequency sweep, to evaluate the ADC frequency response.

1 Brief introduction to ADCs

Analog-to-digital converters, in short ADCs, are electronic devices which translate analog quantities in digital numbers. Usually analog input variables are converted by transducers into voltages or currents. An ADC is defined by three parameters:

N the number of bits used to store each sample;

FSR Full Scale Range, range of the analog values that can be represented by the device;

LSB Least Significant Bit, is the minimum change in the input that guarantees a change in the output, defined as

$$LSB = \frac{FSR}{2^N} \tag{1}$$

The transfer function of an ADC shows the relation between the input and the output, and also shows the quantization error generated by the conversion. A transfer function of an ideal ADC is show in figure 1.

1.1 Parameters explained

DNL stands for Differential Non Linearity, and it is defined as the difference between an actual step width of the transfer function and the LSB. An ideal ADC will thus have

$$DNL_i = 0 \quad \forall i$$

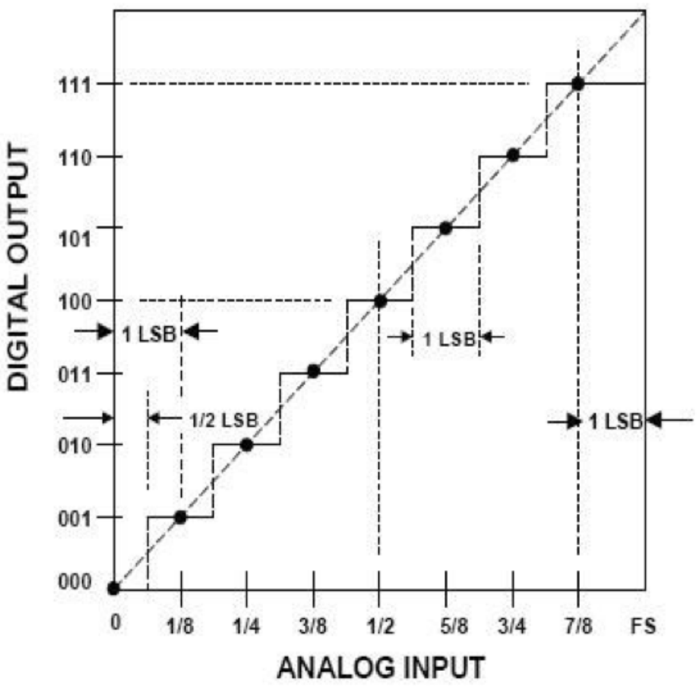

Figure 1: Ideal ADC transfer function

since every step is exactly 1 LSB. The DNL is defined as follow:

$$DNL_i = \frac{f(V_{x+1}) - f(V_x)}{LSB} - 1 \quad i \in]0, 2^N - 2[$$

Remark 1. If the DNL respect the following condition:

$$DNL_i \leq 1\,LSB \quad \forall i$$

then the transfer function $f(x)$ is monotonic, with no missing codes.

INL, Integral Non Linearity, is described as the deviation, in LSB or as a FSR percentage, of an actual transfer function from a straight line. The INL error magnitude depends directly on the position chosen for this straight line. The theoretical formula of measuring INL is the following:

$$INL = \left| \frac{f(V_x) - f(V_0)}{LSB} - x \right| \quad i \in]0, 2^N - 1[$$

The SNR, Signal to Noise Ratio, show how much a signal has been corrupted by noise. The theoretical value of the SNR is defined as follows:

$$SNR_{th} = 6.02 * N + 1.76 \quad [dB]$$

The THD, Total Harmonic Distortion, is defined as the ratio of the signal to the RSS of a specified number of harmonics of the fundamental signal. IEEE Std. 1241-2000 suggests to use the first 10 harmonics. A THD rating $< 1\%$ is desired.

$$THD_n = 10 \log \left(\sum_{i=2}^{n} 10^{\left[\frac{h_i}{20}\right]^2} \right)$$

where h_i is the i-th harmonic expressed in dB.

2 Parameters evaluation formulae and algorithms

2.1 Incoherent sampling

The biggest *problem* we have to account for when evaluating performances of an ADC is incoherent sampling: a sampling is incoherent if

$$\frac{f_s}{f_0} \notin \mathbb{N}$$

Incoherency in sampling will produce the so-called spectral leakage; figure 2 shows how an incoherently sampled sine-wave suffers from spectral leakage:

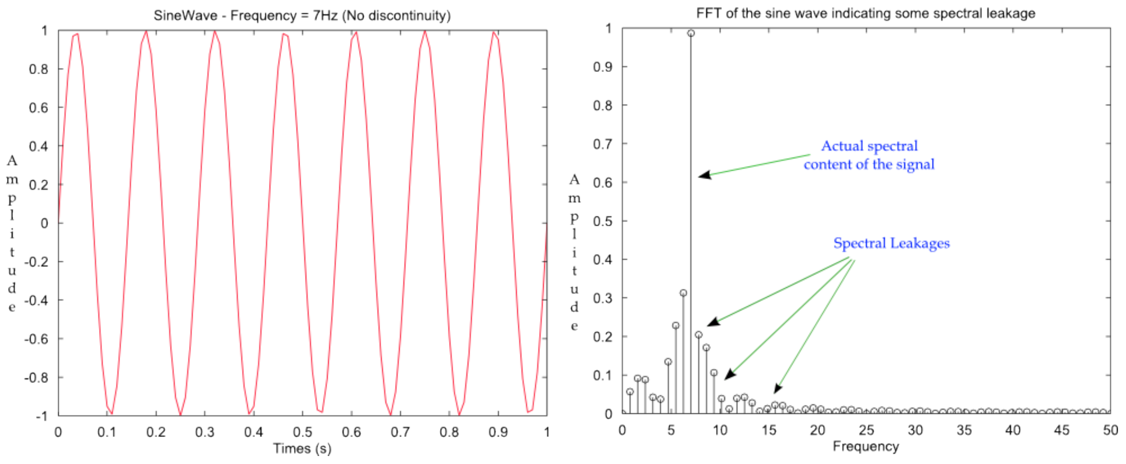

Figure 2: Spectral leakage of an incoherently sampled sine-wave

Since it interferes with all our formulae, we must find a way to *artificially remove* it. The best way so far only works assuming that we have a single tone signal $x[n]$.
First of all we compute the DFT:

$$X[f] = DFT\left\{x[n]\right\}$$

and then we compute:

$$\alpha_1 = \arg\max |X[f]| \tag{2}$$

and:

$$\alpha_2 = \begin{cases} \alpha_1 + 1 & |X[\alpha_1 + 1]| \geq |X[\alpha_1 - 1]| \\ \alpha_1 - 1 & otherwise \end{cases} \tag{3}$$

α_1 is the index of the highest peak in $|X[f]|$, and α_2 is its follower. From these values we can compute the strength of the incoherency:

$$\beta = \frac{M}{\pi} \arctan\left(\frac{\sin\frac{\pi}{M}}{\cos\frac{\pi}{M} + \frac{X[\alpha_1]}{X[\alpha_2]}}\right) \tag{4}$$

The peak lies between α_1 and α_2:

$$\alpha = \alpha_1 + \beta \tag{5}$$

We can then use α as the initial frequency guess for the sinefit4 algorithm, obtaining $\bar{\omega_0}$. Then:

$$\alpha_0 = \frac{\bar{\omega_0}}{\omega_s} M \tag{6}$$

is an improvement over α. We can then use this to trim $x[n]$:

$$x_c[n] = x[n] \quad \forall n = 0..i \tag{7}$$

where:

$$i = \left\lfloor 0.5 + M\frac{\lfloor \alpha_0 \rfloor}{\alpha_0} \right\rfloor$$

We can use $x_c[n]$ to compute all the other parameters; from now on we will refer to $x[n]$ as a coherently sampled signal, unless otherwise specified.

2.2 Histograms

In order to compute DNL and INL in a fast way for a given signal we generate two histograms that count the frequency of all the possible value the ADC can output. The **real histogram** uses the data read from the signal, the **ideal histogram** uses simulated data for the same signal supposing we have a perfect ADC. The signal used in this case is a sinusoid.

The histogram should theoretically have $R_t = 2^N$ bars: for high N values, however, we will have to many bins, slowing down the computation. We fix a resolution limit R_l such that we have:

$$R = \min(R_l, R_t) \tag{8}$$

A good value of R_l could easily be 256 or 512, and it should preferably be a power of two.

2.2.1 Ideal histogram generation

To generate the ideal histogram we need few information:

- the resolution R
- the number of samples M
- the first and last values of the signal, $x[0]$ and $x[M-1]$

We can then evaluate:

$$\phi = \sin\left(\frac{\pi}{2}\frac{M}{M + x[0] + x[M-1]}\right)$$

which will keep the ideal sinusoid we are simulating just below the full scale. We can then define the value of each bar:

$$h_i[n] = \frac{M}{\pi}\left(\arcsin\frac{n-\frac{R}{2}}{\frac{R}{2}} - \arcsin\frac{n-1-\frac{R}{2}}{\frac{R}{2}}\right) \quad \forall n \in [0,\, R-1]$$

2.2.2 Real histogram generation

To generate the real histogram

$$h_r[n] \quad \forall n \in [0,\, R-1]$$

no particular method is needed. For example, *numpy* provides the *hist* function, that calculate an histogram of an array with a given number of bins.

2.3 Non linearities

We use histograms when evaluating DNL and INL in order to speed up the calculation.

2.3.1 DNL

We can easily compute an histogram of the DNL provided we have the ideal and real histograms:

$$DNL[n] = \frac{H_r[n]}{H_i[n]} - 1$$

2.3.2 INL

The INL is evaluated from the DNL:

$$INL[n] = \sum_{i=0}^{n} DNL[i]$$

2.4 Sinusoidal wave frequency detection

In order to compute the SNR of a single tone signal, we need to know the wave's parameters. IEEE proposes two different ways to solve this problem, whether we know or not the wave's frequency.

A wave can represented as a superposition of two waves and a constant term, using five parameters:

$$A\cos(\omega_0 t + \theta) + B\sin(\omega_0 t + \theta) + C$$

The amplitude and the phase of the frequency can then be evaluated this way:

$$\begin{aligned} V &= \sqrt{A^2 + B^2} \\ \theta &= \arctan\frac{B}{A} \end{aligned}$$

2.4.1 sinefit3

This algorithm evaluates A, B, C starting from ω. The algorithm is well described in –INSERT BIBLIOGRAPHY HERE–.

2.4.2 sinefit4

This algorithm evaluates A, B, C and ω_0 starting for an initial approximation of the angular frequency. A good approximation is given by this formula:

$$\omega_i = 2\pi\frac{f_s}{M} arg \max_j X_j$$

where X_j is the DFT of the signal, f_s the sampling frequency and M the number of samples of the signal. Based on sinefit3, this algorithm is essentially a least square fit of the four parameters, and it's well described in –INSERT BIBLIOGRAPHY HERE–.

2.5 Performance parameters

All these parameters are calculated for a digitized sinusoidal wave, eventually multiplied with a window signal.

2.5.1 SNR: Signal to Noise Ratio

The SNR is a rather important parameter, and its general formula is the following:

$$SNR = 20\log_{10}\frac{P_s}{P_n} \quad [dB]$$

with P_s being the signal power and P_n the noise power[1]. This ratio will always be greater than one. The noise power is computed this way:

$$\begin{aligned} C[f] &= \begin{cases} 0 & f \in \{0,\, i\alpha_0,\, M-1-\alpha_0\} \quad \forall i \in [0,\, \left\lfloor \frac{M}{\alpha_0} \right\rfloor] \\ X[f] & otherwise \end{cases} \\ P_n &= \frac{\sum C[f]^2}{M} \end{aligned}$$

2.5.2 SINAD: SIgnal to Noise And Distortion ratio

To compute with great precision this value, we compare our wave with a synthetic one. Using the data obtained from sinefit4 we can generate a signal that represents with great precision (as a perfect ADC) the sinusoid, so we can compute the noise signal:

$$\begin{aligned} s[k] &= C + A\cos(\frac{\omega_0}{f_s}k) + B\sin(\frac{\omega_0}{f_s}k) \ \forall k \in [0,\, M[\\ n[k] &= x[k] - n[k] \end{aligned}$$

At this point we calculate:

$$\begin{aligned} RMS_n &= \sqrt{\frac{1}{M}\sum_{i=0}^{M-1} n[k]^2} \\ RMS_x &= \frac{\max x}{\sqrt{2}} \end{aligned}$$

And then we can define the SINAD:

$$SINAD = 20\log_{10}\frac{RMS_x}{RMS_n}\ [dB]$$

2.5.3 THD: Total Harmonic Distortion

The THS is evaluated in the following way:

$$THD = -20\log_{10}\frac{X[\alpha_0]}{\|h_i\|}$$

where:

$$\begin{aligned} \alpha_0 &= \arg\max X[f] \\ h_i &= X[\alpha_0 i] \quad \forall i \in [2,\, 2-i[\end{aligned}$$

[1]It doesn't take into account the harmonic distortions, in addition to DC and the meaningful signal.

2.5.4 ENOB: Effective Number Of Bits

Let's define the full scale range as:

$$R_{FS} = 2^{N-1}$$

The data range, instead, is the following quantity:

$$R_{data} = \frac{\min x[t] + \max x[t]}{2}$$

Therefore, we define f as follow:

$$f = 20 log \frac{R_{FS}}{R_{data}}$$

in order to avoid penalization for a signal that doesn't cover all the full scale range. The effective number of bits can then be defined as:

$$ENOB = \frac{SINAD + f - 1.76}{6.02}$$

2.5.5 SFDR: Spurious Free Dynamic Range

The spurious free dynamic range is, simply put, the difference between the main peak and the highest harmonic distortion peak in dBc:

$$SFDR_i = 10 \log \frac{\max X[f]}{\max h_i}$$

A good value of i is 10.

3 ACT: ADC Characterization Toolkit

When using this mode, the program will analyze data read either from an ADC or a file. The application only support files at the day of writing.

3.1 Single tone mode

The data supplied to the application is the digitization of a sinusoidal wave. The file structure is rather easy, and contains all the information we need to compute everything:

```
[SIGNAL]
nbits = n
rate = Fs
data =<tab>data[0]
<tab>data[1]
..
<tab>data[M -1]
```

n number of bits of the ADC;

Fs sampling rate;

data a list of integers that represent the signal, always preceded by a TAB character.