Blind Equalization of Static Nonlinearities in SA-ADC

Jonas Elbornsson

Division of Communication Systems
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
WWW: http://www.comsys.isy.liu.se
Email: jonas@isy.liu.se

5th November 2001

Report No.: LiTH-ISY-R-2399
Submitted to CCSSE’01

Technical reports from the Communication Systems group in Linköping are available by anonymous ftp at the address ftp.control.isy.liu.se. This report is contained in the file 2399.pdf.
Abstract

Due to imperfections in the manufacturing process of integrated A/D converters, there are static non-linear errors in the conversion from analog to digital signal. Calibration of these errors is time-consuming and expensive. In this paper a method for blind equalization of the errors is presented, i.e. the errors are estimated from the output signal only. The method has also been evaluated, with some modifications, on measurements from a real A/D converter.

Keywords: A/D conversion, sampling, estimation, blind equalization
Blind Equalization of Static Nonlinearities in SA-ADC

Jonas Elbornsson,
Dept. of Electrical Engineering,
Linköping University, Sweden
E-mail: jonas@isy.liu.se

Abstract

Due to imperfections in the manufacturing process of integrated A/D converters, there are static non-linear errors in the conversion from analog to digital signal. Calibration of these errors is time-consuming and expensive. In this paper a method for blind equalization of the errors is presented, i.e. the errors are estimated from the output signal only. The method has also been evaluated, with some modifications, on measurements from a real A/D converter.

Keywords: A/D conversion, sampling, estimation, blind equalization

1 Introduction

In an analog-to-digital converter (ADC) the requirement on good accuracy in the resistors\(^1\) are normally very high since the accuracy on the digital signal depends on the resistors. When an ADC is implemented on a CMOS chip, the parasite resistances of the wires are used as resistors. The resistances in these wires have very low accuracy, it might vary up to 50 percent from the nominal value on different chips and up to 1 percent on one chip. But an ADC can be made very small and cheap in CMOS, therefore it is interesting to use this technique. To avoid the increasing errors in the digital signal we have to estimate the actual values of the resistances, or equivalently, the actual levels in the ADC. In this report a new method for estimation of the parameters is presented. This method assumes no prior knowledge of the distribution of the input signal, other than smoothness. This method also works online and adapts to changes in the resistances caused by temperature changes for example, that earlier methods have not taken care of [7].

\(^1\)The resistors are used for generating analog reference levels. In many cases, capacitors are used instead, however, the matching problem and the analysis in this report is exactly the same in that case.

2 ADC description

The AD conversion is done by comparing the unknown analog signal to known levels in a resistance ladder [2]. To get an accuracy of \(n\) bits in the digital signal \(2^n\) resistors are required in the ADC, see Figure 1. This type of ADC is called SA-ADC (successive-approximation). The digital value is found by the binary search algorithm [2], where the analog signal is compared to the reference levels. The correct digital value will be the closest reference level smaller than the analog value. Such an \(n\)-bit ADC will give a digital value with \(n\) correct bits as long as the reference levels are correct. If this is the case the transfer function of the ADC, from the analog input to the digital output, will look like the plot to the left in Figure 2. If a rectangularly distributed signal is used as input to this ideal ADC, a histogram of the output will look like the plot to the right in Figure 2. Here the histogram is flat. If there are errors in the reference lev-
els the transfer function will be incorrect as in the plot to the left in 3. Now the histogram will not be flat, as seen in the plot to the right in Figure 3, since the length of the intervals between the levels are different. To correct the histogram a decoder that maps the digital levels onto the correct levels is used. The histogram will now again become flat, but with different lengths of the intervals, see Figure 4.

2.1 Subranging mismatch

To avoid that the resistance ladder becomes too long in a high precision AD converter the subranging technique can be used. Subranging means the several shorter resistance ladders are used in series, see Figure 5. In a two stage subranging ADC with \( m \) bits in the first stage and \( n \) bits in the second stage, the \( m \) most significant bits are descided in the first stage. Then the ground level of the second stage is connected to the correct level in the first stage and the \( n \) least significant bits are descided. An overlap between the first and the second stage is used to enable correction of wrong descisions in the first stage when the second stage is descided.

The subranging technique introduces another kind of static errors. These errors are caused by mismatch between the stages. If the second stage is too long compared to an interval in the first stage, there can be a level in the second stage that does not get any samples, see Figure 6.

3 Method

3.1 Mapping from analog input to digital output

We assume that the ADC spans the signal interval from zero to one, any other range is just a rescaling. The analog input is \( u_t \) and the digital output is \( y_t \). The mapping from analog input to digital output in an ideal ADC is described by

\[
y_t = \lfloor \frac{2^n u_t}{2^n} \rfloor
\]

(1)

where \( \lfloor . \rfloor \) denotes the integer part. In a real ADC there are errors in \( 2^n - 1 \) of the levels (the first level is ground). We use a parameter vector \( \Theta = [\theta_1 \theta_2 \ldots \theta_{2^n-1}]^T \) to describe the errors in the levels. Each \( \theta_i \) is the relative error in the distance between level \( i-1 \) and level \( i \), so the actual reference levels are:

\[
0, \frac{1+\theta_1}{2^n}, \ldots, \frac{k+\sum_{i=1}^{k} \theta_i}{2^n}, \ldots, \frac{2^n-1+\sum_{i=1}^{2^n-1} \theta_i}{2^n}
\]

The analog signal is then mapped onto the nearest lower level.
3.2 Parameter estimation algorithm

Parts of this material has previously been presented in [5]. The estimation algorithm should work online and adapt to possible changes of the parameters. The algorithm therefore has to work with the actual signals that are converted in the ADC. The analog input signal is not measurable (if it was we would not need the ADC). The only assumption that is made about the input signal is that the amplitude distribution is smooth, i.e. the density of adjacent digital values should be quite constant. The parameter estimation is based on statistical properties of the signal. A histogram is calculated from the measured digital values. To estimate the parameters the measured histogram should be compared to a theoretical histogram parameterized with respect to the parameter vector \( \Theta \) [4]. Since the true distribution of the signal is not known we can not calculate the theoretical histogram. Instead we have to estimate it from the measured data. The estimated histogram will be used as the nominal histogram for \( \Theta = 0 \). The estimation of the true histogram is based on the assumption that the distribution function of the input signal is smooth and that the errors in the levels are random. Note:

- We make no temporal assumption on the signal. That is, its spectral content can be anything.
- We only use a smoothness assumption on the signal’s amplitude distribution \( f_U(u) \).
- The amplitude distribution of the measured signal, \( f_Y(y) \), is computed as an approximation of \( f_U(u) \). From the smoothness assumption, we can approximate \( f_U(u) \) by a smoothed version of \( f_Y(y) \) to \( f_U(u) \).
- Next: find a mapping \( y \rightarrow z = g(y, \theta) \) such that \( \hat{f}_Z(z, \theta) \approx f_Y(y) \), where \( \hat{f}_Z(z, 0) = f_U(u) \).

This means that the nominal histogram, \( \hat{f}_Z(k) \), can be estimated by low pass filtering the measured histogram, \( f_Y(k) \). A non-causal zero phase filter should be used to avoid displacement of the histogram. The cut-off frequency in the filter must be chosen for each kind of signal since the frequency content of the true histogram depends on the signal. This choice can be made by studying the Fourier transform of \( f_Y(k) \) where it is usually easy to find the bandwidth of the true histogram. When the parameters are estimated we have to be able to compare the measured histogram with a theoretical histogram for any value of \( \Theta \). This is calculated by interpolation between adjacent histogram bars

\[
\hat{f}_Z(k, \Theta) = \frac{\hat{f}_Z(k, 0) + \hat{f}_Z(k + 1, 0)}{2} + \hat{f}_Z(k, 0) \sum_{i=1}^{k} \theta_i - \frac{\hat{f}_Z(k, 0) + \hat{f}_Z(k - 1, 0)}{2} \sum_{i=1}^{k-1} \theta_i \tag{2}
\]

where \( \hat{f}_Z(j, 0) \) is the low pass filtered measured histogram, \( \hat{f}_Y(j) \). To find an estimate of the parameter vector \( \Theta \) the criterion function

\[
V(\Theta) = \sum_{k=1}^{2^n} (\hat{f}_Y(k) - \hat{f}_Z(k, \Theta))^2 \tag{3}
\]

should be minimized. The search for a minimum can be done in several different ways such as Newton’s method or Gauss-Newton, see for example [6] or [1]. These methods give fast convergence in number of iterations but are numerically sensitive and computationally demanding for large systems since both the first and second derivatives have to be calculated. For hardware implementation it is therefore better to use a steepest descent method, where only the first derivative is calculated. In the steepest descent method the parameters are updated in the negative gradient direction at each iteration

\[
\hat{\Theta}_{i+1} = \hat{\Theta}_i - \eta \nabla V(\Theta) \tag{4}
\]

where \( \eta \) is a constant step size. Both the measured and the estimated histograms are updated with new data at certain time intervals. To make the algorithm adaptable to changes in the parameters an exponential forgetting of the old histograms is introduced. This recursive update method also give automatic validation of the parameter estimate.

\[
(\hat{f}_Y)_{i+1} = \lambda (\hat{f}_Y)_i + (1 - \lambda) \hat{f}_{Y_{\text{new}}}, \lambda \in (0, 1) \tag{5}
\]

\[
(\hat{f}_Z)_{i+1} = \mu (\hat{f}_Z)_i + (1 - \mu) \hat{f}_{Z_{\text{new}}}, \mu \in (0, 1) \tag{6}
\]

This can be summarized into the algorithm:

**Algorithm 1**

**Initialization**

- Choose a cut-off frequency, \( f_{\text{co}} \), for the low pass filter.
- Choose the number of data, \( N \), for each histogram update.
- Choose the number of iterations, \( M \), between histogram updates.
- Choose the forgetting factors, \( \lambda \) and \( \mu \), for the measured and estimated histogram.

**Data collection and histogram update**

- Collect a batch of data of length \( N \) from the ADC and calculate a histogram. The histogram is normalized so that \( \sum_{i=1}^{2^n} \hat{f}_{Y_{\text{new}}}(i) = 1 \).
- Calculate an estimate of the theoretical histogram, \( \hat{f}_{Z_{\text{new}}} \), by low pass filtering the measured histogram.
• Update the the old histograms, \( \hat{f}_Z \) and \( \hat{f}_Y \), with new data:

\[
\begin{align*}
\hat{f}_Z &= \mu \hat{f}_Z + (1 - \mu) \hat{f}_Z^{\text{new}} \\
\hat{f}_Y &= \lambda \hat{f}_Y + (1 - \lambda) \hat{f}_Y^{\text{new}}
\end{align*}
\]

**Iteration**

The following steps are done \( M \) times during the time new data are collected.

1. Calculate an estimate of the histogram for the parameters \( \bar{\Theta} \) by interpolation between adjacent histogram bars:

\[
f_Z(k, \bar{\Theta}) = \hat{f}_Z(k, 0) + \frac{f_Z(k+1, 0) + f_Z(k, 0)}{2} \sum_{i=1}^{k} \hat{\theta}_i
\]

2. Calculate the criterion function:

\[
V(\bar{\Theta}) = \sum_{i=1}^{2^n} (\hat{f}_Y(i) - \hat{f}_Z(i, \bar{\Theta}))^2
\]

3. Calculate the gradient:

\[
(\nabla V(\bar{\Theta}))_m = -2 \sum_{i=1}^{2^n} (\hat{f}_Y(i) - \hat{f}_Z(i, \bar{\Theta})) \frac{\delta f_Z(i, \bar{\Theta})}{\delta \theta_m}
\]

\[
\frac{\delta f_Z(i, \bar{\Theta})}{\delta \theta_m} = \begin{cases} 
 f_Z(i+1, 0) - f_Z(i, 0) & \text{if } m < i \\
 f_Z(i+1, 0) + f_Z(i, 0) & \text{if } m = i \\
 0 & \text{if } m > i 
\end{cases}
\]

4. Initialize the steepest descent step size \( \eta = 1 \)

5. Update the parameter estimate:

\[
\bar{\Theta}_{j+1} = \bar{\Theta}_j - \eta \nabla V(\bar{\Theta}_j)
\]

6. Calculate the criterion function for the new parameter vector:

\[
V(\bar{\Theta}_{j+1}) = \sum_{i=1}^{2^n} (\hat{f}_Y(i) - \hat{f}_Z(i, \bar{\Theta}_{j+1}))^2
\]

7. If \( V(\bar{\Theta}_{j+1}) > V(\bar{\Theta}_j) \) update the stepsize \( \eta = \eta/2 \)

8. Repeat steps 5–7 until the criterion function decreases

Figure 7 shows a block diagram of the AD converter with estimation of the histogram, estimation of the parameters and correction of the output.

**Simulations**

All simulations in this report have been done for an 8-bit AD converter with \( 2^8 = 256 \) levels. The algorithm is evaluated for a Gaussian signal. The Gaussian distribution is chosen so that the mean of the signal lies in the middle of the histogram. The standard deviation is chosen so that the interval \([-5\sigma, 5\sigma]\) falls inside the range of the ADC. In figure 8(a) the theoretical Gaussian histogram is shown. In [3] the ADC is also simulated with sinusoid input.

**Figure 7. Block diagram of the whole correction model.**

### 4.1 Performance

The parameter estimation has been done as described in section 3 with \( 10^5 \) samples of data and \( 10^3 \) iterations and no forgetting. The errors in the reference levels are generated randomly, so that the relative error in the levels are uniformly distributed in the interval \([-0.1, 0.1]\), i.e. the error between two consecutive reference levels is at most 10 percent.

#### 4.1.1 Histogram for compensated ADC

One interesting performance measure is how close the measured histogram compensated by the estimated parameters comes to the theoretical histogram. The compensated histogram is normalized with the width of each histogram bar so that it is comparable with the true histogram. In Figure 8 four histograms for Gaussian input signals are shown. Figure 8(a) shows the theoretical Gaussian histogram that the measured histogram would converge to with all errors equal to zero and infinite number of data. Figure 8(b) shows the measured histogram. Figure 8(c) shows the estimated amplitude distribution of the input signal, \( f_Z(k, 0) \). Figure 8(d) shows the histogram of the output signal compensated with the estimated error parameters. In these plots we can see that the estimation works quite well. To get a better view of how large the errors are the relative error between the theoretical and the compensated histogram is shown in Figure 9(a). Here we can see that the estimation works well, especially in the middle of the histogram.

#### 4.1.2 Evaluation of the histogram estimate

To find out if the errors in Figure 9(a) comes from the parameter estimation or from the histogram estimation we make a plot of the relative error between the estimated histogram and the theoretical histogram.
4.2 Evaluation of the parameter estimates

4.2.1 Distortion measures

In this section we will evaluate two different measures of the distortion caused by the errors in the ADC [8], the SNDR (signal to noise plus distortion ratio) (7) and SFDR (8) (spurious free dynamic range).

\[
SNDR = 10 \log \frac{E(signal)}{E(noise + distortion)} dB \tag{7}
\]

\[
SFDR = 10 \log \frac{E(signal)}{E(largest dist. component)} dB \tag{8}
\]

where \( E(\cdot) \) is the energy. The second column in table 1 shows the SNDR values for the output signal corrected with different estimates of \( \Theta \). The SNDR for the output without correction is also included for comparison. The SNDR is measured with a sinusoid input signal. These values show

<table>
<thead>
<tr>
<th>( \Theta ) estimate</th>
<th>SNDR</th>
<th>SFDR</th>
<th>RMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>no correction</td>
<td>40.7 dB</td>
<td>51 dB</td>
<td>1.01</td>
</tr>
<tr>
<td>True ( \Theta )</td>
<td>48.5 dB</td>
<td>72 dB</td>
<td>0</td>
</tr>
<tr>
<td>( \Theta ) estimated from estimated Gaussian histogram</td>
<td>48.3 dB</td>
<td>61 dB</td>
<td>0.12</td>
</tr>
<tr>
<td>( \Theta ) estimated from known Gaussian histogram</td>
<td>48.7 dB</td>
<td>65 dB</td>
<td>0.10</td>
</tr>
</tbody>
</table>

Table 1. SNDR and SFDR with sinusoid input and RMS for different estimates of \( \Theta \)

that for a Gaussian input signal we reach optimal performance measured in SNDR even if we do not know the distribution. The SNDR gives a measure of the mean performance. If there are just a few distortion frequencies the SNDR can still be quite good. Therefore we will also look at the SFDR, that gives a worst case performance measure. The third column in table 1 shows the SFDR values for different estimates of \( \Theta \).

4.2.2 RMS error

The RMS (root mean square) error measures the mean error of the levels in the ADC:

\[
RMS = \sqrt{\frac{1}{2^n - 1} \sum_{i=1}^{2^n-1} \left( \sum_{j=1}^{i} (\theta_j - \hat{\theta}) \right)^2} \tag{9}
\]

The RMS values for the different estimates described in section 4.2.1 are shown in the last column in table 1. From this
table we can see that the RMS error is lower for all estimates of \( \Theta \) but not always very much better. The reason for this is that the estimates near the edges of the ADC are quite bad and therefore they give a large contribution to the rms, the estimates in the middle of the ADC is much better.

5 Measurements

The estimation algorithm is in this section evaluated on measurements from a real ADC. A three stage subranging ADC with 4, 4, 5 bits respectively was used. The algorithm is here modified to handle only large errors. This is done to reduce the calculation complexity and the required amount of data.

5.1 Data acquisition

A sequence of Discrete Multi Tone (DMT) symbols was used as input signal to the ADC when calculating the histogram. In Fig. 10, an example with \( 5 \cdot 10^6 \) samples is shown.

5.2 Evaluation

The correction algorithm have been evaluated with sinusoidal signals. The signal power has been varied and the Signal to Noise and Distortion Ratio (SNDR) and Spurious Free Dynamic Range (SFDR) has been measured before and after correction for each signal, Fig. 11. The peak improvement after correction is about 8dB for SNDR and 14dB for SFDR.

6 Conclusions

A method for blind equalization of the static non-linear errors in an AD converter has been developed. The method does not assume any prior knowledge of the signal except that the amplitude distribution should be smooth. The method has been evaluated in simulations for Gaussian and sinusoid input signals and the estimation works especially well for Gaussian input. The reason why it does not work so well for a sinusoid input is that the distribution contains a quite sharp knee that is removed in the estimation. The method has also been evaluated on measurements from a real ADC with some simplifications. These measurements show a good improvement of the signal quality.

References


