ADC Modeling for System Simulation

Kalle Folkesson

LiU-TEK-LIC-2003:26
Department of Electrical Engineering
Linköpings universitet, SE-581 83 Linköping, Sweden
Linköping 2003

ISBN 91-7373-689-9
ISSN 0280-7971
Abstract

Modern system design methods are based on early system simulation using behavioral models. Since the ADC often is the critical component it is especially important that it is modeled correctly. Most current design environments use very simple ADC models such as ideal sampling and quantization to give a certain effective number of bits (ENOB). The ENOB is typically found by a single-tone test where the input is an amplitude-limited sine wave. Real applications may have inputs very different from this simple test signal and the ADC can then have a completely different performance. Detailed knowledge of the behavior in a system allows the ADC design margin to be minimized thus saving cost and power consumption.

In the work included in this thesis an accurate model of a successive-approximation ADC is developed. It is aimed for integration into existing system simulators and can thus not be too complex or the simulation time will be unreasonably long. Measurements are performed to validate the model and comparisons between simulated and measured data are done in both time and frequency domains. This is used to test some error hypotheses in order to find the performance limiting errors of the successive-approximation architecture.

Further, the need for accurate ADC models in system simulation is investigated. A complex model is of no use if the same information can be obtained with a simple one. To do this, the ADC model is integrated in two different systems: an ADSL modem and a radar receiver. System simulations are performed and the results are compared to the case when a simple ADC model is used. In both cases the accurate ADC model
showed to be useful. System performance varied quite much for ADCs with the same specified performance in terms of ENOB depending on which error mechanism was active.
Preface

This thesis presents the part of my research at the Electronic Devices group, Department of Electrical Engineering at Linköping University from December 1998 to December 2002 that concerns ADC modeling. The following papers are included:


Related, but not included papers are:

• S. Brodén, M. Danestig, K. Folkesson, H. Ohlsson, B. Svensson, and A. Åström, "Smart Sensors", in Proceedings of RVK99, Jun. 1999


Acknowledgements

I would like to thank the following people

- My supervisor Professor Christer Svensson for his guidance, support, and patience.
- My co-supervisor during the first years, Dr. Jan-Erik Eklund.
- Henrik Ohlsson, Andreas Gustafsson, and Dr. Jonas Elbornsson for interesting cooperation.
- Roland Standert for a job well done on specification of radar applications.
- Rutger Carlsson and Arta Alvandpour for fixing problems related to tools and computers. Thank you also Mattias Arvidsson for showing me a slightly less stupid method to insert pictures in Word.
- Ingegärd Andersson and Anna Folkeson for help with administrative stuff.
- A big thank you goes to Lic. Eng. Daniel Eckerbert for miscellaneous technical support though he has more than enough
work of his own, not to mention everything he does for other people...


- I would like to thank all past and present members of the Electronic Devices group for creating a nice working environment, especially Stefan Andersson and Peter Caputa, Lic. Eng. Ulf Nordquist, Dr. Tomas Henriksson, and Professor Dake Liu, and Professor Per Larsson-Edefors.

- The Swedish Foundation for Strategic Research (SSF) for sponsoring this work through the Smart Sensors project.

Kalle Folkesson
## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A/D</td>
<td>Analog-to-Digital</td>
</tr>
<tr>
<td>ADC</td>
<td>Analog-to-Digital Converter</td>
</tr>
<tr>
<td>ADSL</td>
<td>Asymmetric Digital Subscriber Line</td>
</tr>
<tr>
<td>DAC</td>
<td>Digital-to-Analog Converter</td>
</tr>
<tr>
<td>DMT</td>
<td>Discrete Multi-Tone</td>
</tr>
<tr>
<td>DNL</td>
<td>Differential Nonlinearity</td>
</tr>
<tr>
<td>ENOB</td>
<td>Effective Number Of Bits</td>
</tr>
<tr>
<td>FMCW</td>
<td>Frequency-Modulated Continuous Wave</td>
</tr>
<tr>
<td>INL</td>
<td>Integral Nonlinearity</td>
</tr>
<tr>
<td>LSB</td>
<td>Least Significant Bit</td>
</tr>
<tr>
<td>MSB</td>
<td>Most Significant Bit</td>
</tr>
<tr>
<td>OFDM</td>
<td>Orthogonal Frequency Division Multiplex</td>
</tr>
<tr>
<td>QAM</td>
<td>Quadrature Amplitude Modulation</td>
</tr>
<tr>
<td>SA-ADC</td>
<td>Successive-Approximation Analog-to-Digital Converter</td>
</tr>
<tr>
<td>SFDR</td>
<td>Spurious-Free Dynamic Range</td>
</tr>
<tr>
<td>SNDR</td>
<td>Signal-to-Noise-and-Distortion-Ratio</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal-to-Noise-Ratio</td>
</tr>
<tr>
<td>T/H</td>
<td>Track-and-Hold</td>
</tr>
<tr>
<td>THD</td>
<td>Total Harmonic Distortion</td>
</tr>
<tr>
<td>VLSI</td>
<td>Very Large Scale Integration</td>
</tr>
</tbody>
</table>
Contents

Abstract ........................................................................................................ iii
Preface ........................................................................................................ v
Acknowledgements ................................................................................... vii
Abbreviations ........................................................................................... ix

I Introduction ......................................................................................... 1

1 Introduction ....................................................................................... 3
  1.1 Background ................................................................................... 3
  1.2 ADC Applications ......................................................................... 4
    1.2.1 ADSL ................................................................................... 4
    1.2.2 Radar ................................................................................... 5
  1.3 Contributions ................................................................................ 5
  1.4 References .................................................................................... 6

2 ADC Architectures ............................................................................... 9
  2.1 Introduction ................................................................................... 9
  2.2 Flash ............................................................................................. 9
  2.3 Pipelined ....................................................................................... 10
  2.4 Successive-Approximation ......................................................... 11
  2.5 Integrating .................................................................................. 12
  2.6 Sigma-Delta ................................................................................. 13
10.3.1 Simulation Setup ............................................................ 80
10.3.2 Simulation Results .......................................................... 81
10.4 Conclusions ........................................................................... 83
10.5 References ............................................................................. 84

11 Paper 5 ....................................................................................... 85
11.1 Introduction ............................................................................. 86
11.2 Model Descriptions ............................................................... 87
  11.2.1 Radar Model ................................................................. 87
  11.2.2 ADC Model ................................................................. 88
11.3 Simulations ............................................................................. 90
  11.3.1 Simulation Setup ............................................................ 90
  11.3.2 Simulation Results .......................................................... 90
11.4 Conclusions ........................................................................... 92
11.5 References ............................................................................. 93

IV Appendix .................................................................................... 95

A ADC Equations ........................................................................... 97
  A.1 SNR ................................................................................. 97
  A.2 Jitter ................................................................................. 99

B Model Equations ........................................................................ 101
  B.1 Calculate Settled Reference Voltage .......................... 102
  B.2 Calculate Settled Sampled Input Voltage ................... 104
  B.3 Calculate Comparator Output Voltage ...................... 106
Part I

Introduction
1

Introduction

1.1 Background

Analog-to-digital converters (ADCs) are key components in signal processing systems such as communication applications and radar. As the advances in VLSI technologies allow more and more circuitry to be integrated on one chip, the ADCs become just cells in more complex circuits or systems on chip. Hence, testing stand-alone ADCs is becoming less relevant [1]. Also in simulation, the ADC should be put in an application.

Modern design methods are based on early system simulation using behavioral models [2]. It is practical to simulate analog parts in frequency domain and digital parts in time domain but most current design environments focus on one of these and thus cannot efficiently handle both analog and digital parts. The ADC models included in system simulators are often very simple such as ideal sampling and quantization to give a certain effective number of bits (ENOB). Real ADCs on the other hand have many errors apart from the quantization error [3]. Another issue is that the ENOB number used for characterization is found by performing a single-tone test [4]. In this test, the output is analyzed when the input is a full-scale sine wave. For other inputs, the performance can be very different so the specified ENOB is not valid for an arbitrary application [5]. ADCs normally have a significant impact on system performance and to ensure that the system will perform according to the specifications, the ADC is often over-
specified to compensate for errors not included in the model. This makes the system unnecessarily complex and expensive.

One method to find suitable ADC requirements and thereby avoid costly over-specification is to perform system simulations using an accurate ADC model which includes all performance limiting errors. It is also interesting to perform simulations to find out which error mechanism is performance limiting in a certain application. Errors in ADCs are architecture dependent so with this strategy suitable architectures can be found as well as knowledge on where to put the design effort. Deterministic errors can easily be corrected, but random ones are more difficult to handle. Since some mechanisms, such as mismatch, can cause either deterministic or random errors depending on the application, this is also useful to simulate.

1.2 ADC Applications

In mixed-signal design it is desirable to implement as much as possible in the digital part since this gives more efficient signal processing and integration as well as increased flexibility. Because of this, functions that traditionally have been performed in the analog domain are moved to the digital. If a function of an analog component that relaxes the ADC requirements, e.g. a mixer, is moved to the digital domain, the ADC requirements will, of course, increase. Because of this, the ADC often becomes the bottleneck.

As discussed in the previous section, ADCs are normally specified in terms of ENOB, which is related to a specific input. It is therefore of interest to investigate applications with other types of inputs to see how the ENOB characterization holds. To be relevant, it should give the same system performance no matter which error mechanism is active in the ADC. Input signals and important ADC properties for applications used in this thesis are briefly presented in the following sections. The applications are presented with a little more detail in the papers they are used; section 10 for ADSL and section 11 for radar.

1.2.1 ADSL

In an ADSL application, the ADC input is a discrete multi-tone (DMT) signal. It consists of 256 sine waves with different modulation. These 256 channels are individually QAM modulated with 2 to 15 bits depending on the signal-to-noise-and-distortion ratio (SNDR) of each
channel. The higher the SNDR in a channel, the more bits will be assigned to it. This way, all channels can be fully utilized. Since any noise or distortion will degrade the system performance, the most important ADC characteristic for this application is SNDR.

### 1.2.2 Radar

The ADC input in a frequency-modulated continuous wave (FMCW) radar application consists of multiple sine waves representing different targets. The signal power levels vary quite much depending on the structure of and distance to the corresponding target. The combination of multiple signals and the possible large difference in their power levels sets very tough requirements on the ADC. If a signal has a large enough power, it is recognized as a target. Spurious signals, such as harmonic distortion, will thus appear as false targets if their power is above the detection threshold. For this type of radar, the spurious-free dynamic range (SFDR) limits the performance since it is a measure on how well it can detect weak signals in the presence of strong interfering signals such as closer targets. Noise is not a big problem as long as it is below the detection threshold. Hence, the most important ADC characteristic is SFDR.

### 1.3 Contributions

The main contributions of this thesis are:

- Development and validation of an accurate model of a successive-approximation ADC (SA-ADC) aimed for system simulation.

- Motivating the use of accurate ADC models in system simulation.

This section gives a short summary of the contents of each paper.

Uncertainty in sampling instant, jitter, is one of the most important dynamic limitations [3]. For high input frequencies it will limit the performance of an ADC. In the first paper [6], an ADC model to model jitter behavior is developed. It is validated by comparisons to measured data and its integration in an RF system simulator, Agilent ADS, is demonstrated.
In the second paper [7], the model is expanded to include more errors and is more thoroughly validated with measurements in both time and frequency domains. The model was used to investigate the importance of various dynamic error mechanisms in SA-ADCs.

The successive-approximation architecture suffers from low throughput and to be of use in high-frequency applications, a time-interleaved structure has to be used. Interleaving, however, introduces errors and to keep the performance these have to be corrected. Therefore, to motivate the use of SA-ADCs, some work on error correction of time-interleaved structures is included. To support Jonas Elbornsson in the validation of his error estimation algorithm developed in [8], some measurements on time-interleaved ADCs were performed. The results presented in [9] shows that the signal quality is improved after correction based on estimates using this method.

The detailed knowledge of dominating error mechanisms in SA-ADCs gained in [7] was then used to investigate their effects on system behavior [10], [11]. For an ADSL application, [10], and a radar application, [11] it was shown that ADCs with the same specified performance in terms of ENOB gave very different system performance depending on which error mechanism was dominating in the ADC. These results motivate the use of accurate ADC models in system simulation.

1.4 References

1.4 References


ADC Architectures

2.1 Introduction
An ADC produces a digital code to represent an analog input. The analog-to-digital (A/D) conversion is always based on comparisons between the analog input signal and known reference levels but there are many different approaches to perform these, each with its own advantages and disadvantages. In this section the most common ADC architectures are presented in their simplest form. More information can be found in [1]-[7]. Some examples of commercially available state-of-the-art ADCs are presented in section 2.9.

2.2 Flash
The flash architecture is the fastest method to convert an analog signal to a digital one. All output bits are calculated in one clock cycle using parallel comparators as shown in figure 2.1. A voltage divider is used to generate the reference voltages for the comparators. The comparator outputs will be ‘1’ up to the reference closest below the analog input and then ‘0’ thus creating a thermometer code. This is then decoded to the digital output code. An \( n \)-bit converter requires \( 2^n - 1 \) comparators. This means an exponential growth in size and power dissipation for increasing number of bits. The component matching requirements also double for
every additional bit, which limits the useful resolution of a flash converter to 8-10 bits. Low-resolution converters can achieve sample rates of a few GS/s. The flash is typically used in high-frequency applications that cannot be addressed any other way and where high precision is not very important. Examples are point-to-point radio links and sampling oscilloscopes.

### 2.3 Pipelined

The pipelined architecture overcomes the limiting factors of the flash by dividing the conversion task into several stages. It consists of a number of stages each including track-and-hold (T/H), low-resolution ADC and DAC, summing circuit, and an amplifier to provide inter-stage gain. A block diagram is shown in figure 2.2. If 1-bit converters are used, the first stage is a coarse 1-bit ADC that calculates the MSB. The MSB is then converted back to an analog voltage by the 1-bit DAC. The difference between the analog input and the analog representation of the MSB is sent to the next stage in the pipeline after a multiplication by two to compensate for the change in significance level.

Using the pipelined architecture, it is possible to achieve higher resolution than with the flash but the total conversion time is increased to $m$ clock cycles for an $m$-stage pipeline. However, since $m$ samples are
processed simultaneously, the total throughput is the same as for a flash. The difference is only a latency of \( m \) cycles. Also, due to the more complex design that requires a longer settling time, it is not possible to design it to be as fast as a flash. The pipelined ADC has a good balance between size, speed, resolution, and power dissipation and has therefore become the most popular architecture for applications where a high sampling rate as well as high resolution is needed, e.g. digital video and communication systems such as radio base stations and ADSL modems. The resolution typically ranges from 8 to 16 bits with sampling rates of a few MS/s for the higher resolutions and up to a couple of hundred MS/s for the lower.

2.4 Successive-Approximation

While the flash converter uses many comparators to do the conversion in one clock cycle, the successive-approximation converter (SA-ADC) does the exact opposite. Here, the conversion is performed using one single converter in many clock cycles to make the successive approximations of a binary search. The block diagram is shown in figure 2.3. Basically it consists of a DAC in a feedback loop. For each step in the binary search, a comparison is made between the input and a reference level. The logic block updates the output register depending on the outcome of the comparison and selects the appropriate reference level to be generated by the DAC for the comparison in the next significance level. An \( n \)-bit conversion is performed in \( n \) clock cycles and, as for the flash and pipelined converters, the component matching requirements doubles for
each additional bit. The resolution typically ranges from 8 to 18 bits at sampling speeds up to 5 MS/s.

Since the latency of the SA-ADC is only one sampling cycle (however many cycles of the internal clock), a conversion can be started at any time. This makes the SA-ADC suitable for applications with non-periodic inputs. It is ideal to convert multiplexed signals. For e.g. a pipelined architecture, which has longer latency, a delay of at least the latency must be added to avoid interference of the multiplexed signals. The suitability to convert multiplexed signals is crucial also for simultaneous conversion of multiple signals in one ADC. This is necessary if there is phase information between different channels, as e.g. for I and Q channels. The simultaneous conversion can be performed by using multiple T/H to sample the signals at the same time instant and then use a multiplexing scheme for the quantization.

The low latency also allows the SA-ADC to be turned off when no conversion is needed. Hence, the power dissipation scales with the sample rate, while it for e.g. pipelined architectures normally is constant. This makes the SA-ADC useful in low-power applications where the data acquisition is not continuous e.g. PDAs (Personal Digital Assistants).

### 2.5 Integrating

A block diagram of an integrating converter is shown in figure 2.4. The input signal is integrated and compared to a known reference level. A counter counts the number of clock cycles it takes until the input reaches the reference level and the comparator output switches. This time is proportional to the input voltage. The problem with this approach is that it is dependent on the tolerances of the R and C values of the integrator. To solve this problem a dual-slope architecture can be used. Then the integrator has switched inputs and charges with the input for a known time, $t_{\text{charge}}$, and discharges with a known opposite-polarity reference.
voltage, $V_{\text{ref}}$, until it reaches zero. The time for discharge, $t_{\text{discharge}}$, is measures and the input can be calculated as

$$V_{\text{in}} = V_{\text{ref}} \frac{t_{\text{charge}}}{t_{\text{discharge}}}.$$  

With this technique, any error introduced by component value imperfection will be canceled out during the discharge. The dual-slope converter has no problems with exponential increase of size and component matching requirements. The circuitry will not change for increasing resolution. The conversion time, however, increases exponentially. To double the resolution, the resolution of the time measurements has to double, which requires integration over twice as many clock cycles. An interesting feature of the integrating architecture is the ability to reject unwanted signals. Using an integrate cycle of $T$, all frequencies of $n \cdot \frac{1}{T}$ will integrate to zero and thus be completely rejected. The integration time can thus be chosen to reject unwanted frequencies.

Integrating ADCs typically achieves resolutions of 12 – 16 bits, but are very slow. The sample rate is only about 100 S/s. Typical applications are instrumentation, e.g. digital multimeters. Due to its good noise rejection it is useful in noisy industrial environments. Examples are digitizing outputs of strain gauges and thermocouples.

### 2.6 Sigma-Delta

The block diagram of a sigma-delta converter is shown in figure 2.5. The output from a 1-bit DAC is subtracted from the input signal. The resulting signal is integrated and then fed to a comparator, which converts it to a 1-bit digital output. This is then used as input to the DAC,
which makes sure that the average output from the integrator is close to
the reference level of the comparator. This loop is run at an oversampled
rate, much faster than the sample rate, and it produces a stream of ones
and zeros from the comparator output. The density of ones in the data
stream is proportional the input signal value. A digital filter performs
low-pass filtering and decimation of the data stream and generates the
digital output code. For an $n$-bit sigma-delta converter, a simple
implementation of the digital filter is an $n$-bit counter. In real converters,
however, the filters are much more advanced. One commonly used
topology is the sinc$^3$.

The sigma-delta makes use of two principles to obtain high resolution:
oversampling and noise shaping. Oversampling reduces the quantization
noise by means of increasing the noise bandwidth. Another advantage
with the oversampling is that it relaxes the requirements for an external
anti-aliasing filter. By summing the error voltage, the integrator has a
noise shaping feature. It acts as a low-pass filter for the input signal and
as a high-pass filter for noise thus moving this up in frequency and out of
the used band. The oversampled noise-shaper effectively trades speed for
accuracy. This means that the sigma-delta does not have the tough
requirements on component matching as the other architectures. High-
resolution converters can be designed without the need for precision
analog elements, which allow the sigma-deltas to be implemented in
standard digital processes. This also gives it low power dissipation.
Typical resolution is 12 – 24 bits. The high oversampling ratio needed to
obtain this limits the sampling rate to about 100 kHz. Like pipelined
converters, the sigma-deltas have latency. Sigma-delta converters are
used in low-frequency applications with high resolution requirements
such as voice-band communication and audio.

Figure 2.5. Sigma-delta ADC block diagram.
2.7 Subranging

Generating many reference levels separately requires many resistors, one per reference level, and will use much area. To save area, subranging can be used. This means that the conversion is divided into multiple steps. In the first step a coarse conversion is performed producing the MSBs. The difference between the input and the analog representation of the MSBs is then passed to the next stage where another conversion is performed. If two subranging stages with $n_1$ and $n_2$ reference levels respectively are used, only $n_1 + n_2$ resistors are needed to produce the equivalence of $n_1 \cdot n_2$ reference levels. Subranging is a part of the pipelined architecture, but it can also easily be implemented in SA-ADCs. If the subranges are chosen to overlap, there will be a redundancy that can be used for error correction.

2.8 Interleaving

The sampling rate of a system can be increased by using several time-interleaved ADCs [8]. A higher sampling rate is obtained at the cost of more hardware by running the ADCs in parallel, but at different clock phases, see figure 2.6. Thereby the requirements on each ADC is relaxed.
since the time available for conversion is increased by a factor of $m - 1$, where $m$ is the number of time-interleaved AD\textsuperscript{C}s. The sampling, however, still has to be performed at full speed. Ideally, $m$ time-interleaved converters will have the same performance as one converter but at $m$ times higher sampling frequency. In a real case, there are some problems that arise when time-interleaved AD\textsuperscript{C}s are used. Mismatch in gain, offset, and phase between the time-interleaved channels will cause errors that have to be corrected to make full use of the gained performance. To be able to time interleave AD\textsuperscript{C}s, a division of the clock signal is also necessary.

### 2.9 Summary

Figure 2.7 summarizes the relative performance in resolution and speed for the discussed architectures and some examples of state-of-the-art AD\textsuperscript{C}s are shown in table 2.1.
Table 2.1. State-of-the-art ADCs.

<table>
<thead>
<tr>
<th>Number of Bits</th>
<th>Sampling Rate [MS/s]</th>
<th>Architecture</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>1500</td>
<td>flash</td>
<td>Maxim [9]</td>
</tr>
<tr>
<td>10</td>
<td>210</td>
<td>flash</td>
<td>Analog Devices[10]</td>
</tr>
<tr>
<td>12</td>
<td>210</td>
<td>pipelined</td>
<td>Analog Devices[11]</td>
</tr>
<tr>
<td>14</td>
<td>80</td>
<td>pipelined</td>
<td>Analog Devices[12]</td>
</tr>
<tr>
<td>18</td>
<td>0.8</td>
<td>SA-ADC</td>
<td>Analog Devices[13]</td>
</tr>
<tr>
<td>24</td>
<td>0.096</td>
<td>Sigma-delta</td>
<td>Analog Devices[14]</td>
</tr>
</tbody>
</table>

2.10 References


3

ADC Characterization

3.1 Introduction
There are many parameters for ADC characterization, describing different aspects of the conversion. They are all important for different reasons. In the following sections, the most commonly used specifications are defined. More information can be found in [1] and [2].

3.2 DC Specifications
When performing an A/D conversion, there are several mechanisms that limit how accurately the signal is represented.

When converting an analog signal to a digital, there is a round-off error. This quantization error exists even in ideal ADCs and sets a theoretical upper limit on achievable resolution. It is a deterministic error, but since the input to an ADC typically is complicated signals and noise, it is randomized. For this reason, quantization error is normally treated as white noise.

Besides the quantization noise, thermal noise will reduce the resolution. It is a fundamental random noise that is present in all systems.

Another factor that affects the output code is mismatch from chip manufacturing. The matching can be improved with special design methods or by moving to a process with smaller feature size.
3.2.1 INL
Integral nonlinearity (INL) is a measure of how far from the ideal transfer curve a measured converter result is, see figure 3.1.

3.2.2 DNL
The differential nonlinearity (DNL) is a measure of how far a code is from a neighboring code. The distance is measured as a change in input voltage amplitude and then converted to LSBs. This is illustrated in figure 3.2. A DNL of <±1 LSB ensures that there are no missing codes. INL is the integral of DNL, so a good INL guarantees a good DNL.

3.3 Dynamic Specifications
Having good values on INL and DNL does not necessarily mean that a converter will perform well. Those measures are only valid at or near DC. As the frequencies increase, the ADC performance will decrease due to various dynamic effects.

Jitter, uncertainty in sampling instant due to ADC or sampling clock imperfections, will limit the performance for high input frequencies. Details on the effect of jitter for high input frequencies can be found in appendix A.2.

There will also be dynamic effects due to time constants in T/H, DAC, and comparator. The RC settling of these components sets a limit for maximum clock frequency. There will be a performance decrease when the ADC is run too fast for the sampling capacitors to charge correctly or too fast for the comparator to make a correct decision.
3.3 Dynamic Specifications

3.3.1 SNR
The signal-to-noise-ratio describes where the noise floor is by relating the signal power to the noise power.

\[
SNR = 10 \log \left( \frac{P_s}{P_n} \right) \text{ [dB]},
\]

where \( P_s \) is the signal power and \( P_n \) the total noise power. In an ideal converter, the only noise is the quantization noise and, if the input is a full-scale signal, the SNR can be calculated as

\[
SNR = 6.02n + 1.76 \text{ [dB]},
\]

where \( n \) is the number of bits. See figure 3.3 for an illustration and appendix A.1 for details. In real applications, there are of course more noise contributions and the SNR is lower, but this sets the theoretical maximum SNR.

3.3.2 SNDR
It is not only noise that degrades the performance of ADCs. There is also distortion. The signal-to-noise-and-distortion-ratio (SNDR) is defined as

\[
SNDR = 10 \log \left( \frac{P_s}{P_{n+d}} \right) \text{ [dB]},
\]

where \( P_s \) is the signal power and \( P_{n+d} \) is the total power of noise and distortion.

3.3.3 ENOB
An ADC is designed to have a certain nominal number of bits, a resolution directly related to the number of quantization levels available. In a real ADC this resolution is never achieved since there are always more noise sources than just the quantization noise. As shown in section 3.3.1, the maximum SNR can be calculated from nominal number of bits. For a certain SNDR an equivalent resolution, the effective number of bits (ENOB), for the ADC can be defined as
3.3.4 SFDR

The spurious-free dynamic range (SFDR) gives a measure of how well an ADC can convert weak signals. It is defined as

$$SFDR = 10\log \left( \frac{P_s}{P_{d,\text{max}}} \right) \text{ [dB]},$$

where $P_s$ is the signal power and $P_{d,\text{max}}$ is the power of the strongest spurious peak. This is illustrated in figure 3.3.

3.3.5 THD

The total harmonic distortion is defined as

$$THD = 10\log \left( \frac{P_s}{P_{d,\text{harm}}} \right) \text{ [dB]},$$

where $P_s$ is the signal power and $P_{d,\text{harm}}$ is the power of all harmonic distortion.
3.4 References


4 Error Correction

4.1 Error Correction

It is common that the performance of ADCs is improved by error correction in the digital domain. This can be done by adding correction terms from a look-up table, see further section 5. This is easily done as long as good estimates of the errors are available. The problem is to find these good estimates. It can be done with calibration using a known input signal but this is time consuming and expensive. A lot can be saved if a technique that does not require any knowledge of the input is used to calibrate the ADC at runtime. Error correction is used for single ADCs but is even more important for time-interleaved structures, where additional errors are introduced. In the next section, the errors that arise in time-interleaved structures are described and suggestions are made on how they can be corrected.

4.2 Correction of Time-Interleaved Structures

As discussed in section 2.8, time-interleaving can be used to increase the sampling rate for a certain resolution. Interleaving, however, introduces errors, which must be corrected to achieve this. Due to mismatch in gain, offset, and timing (phase) between the ADCs, the signal will be distorted.
Offset mismatch will distort the signal every time a sample is taken using an ADC with different offset than the one used to take the previous sample. This means that offset errors will show up in the spectrum as a peak at the sample frequency for the individual ADCs, $f_{s,i}$, [1]. For a case with two interleaved ADCs, this is illustrated in figure 4.1. Offset mismatch can be measured by sending signals with the same number of periods to the ADCs and then study the average values. This information can then be used to correct the offset errors.

Gain and timing offset mismatch will have the same effect on the signal. It is not possible to determine if the signal is distorted by gain or timing errors just by observing it. One method to find out is to vary the input frequency. Gain errors will not vary with input frequency, but timing errors will increase linearly as the sampling frequency increases. Gain and timing offset errors will affect the signal with a frequency that is $f_{s,i}$ modulated with the input frequency, $f_{in}$. After folding, it will appear in the spectrum as peaks at $f_{s,i} - f_{in}$ [1], see figure 4.2 and 4.3. One method to find the gain mismatch between interleaved ADCs on the same chip is to add constant voltage generators and thereby be able to study the gain of each ADC. This information can then be used to correct gain errors. Time errors, however, are more difficult to correct. A calibration technique to minimize the time errors is presented in [2]. Depending on the input frequency it improves the SFDR by 20-60 dB. This technique, however, requires a known calibration signal. An algorithm to estimate timing offset errors without any knowledge of the input signal is presented in [3], [4]. It is based on the basic idea is that signals change more in average if the sampling instant is delayed and less if it comes too early. This improvement with this technique is good for low frequencies and tends to zero near the Nyquist frequency.

### 4.3 References

Figure 4.1. Offset error in a) time and b) frequency domain.

Figure 4.2. Gain error in a) time and b) frequency domain.

Figure 4.3. Timing error in a) time and b) frequency domain.


Part II

ADC Modeling
5

ADC Modeling Survey

5.1 ADC Modeling

ADC models are used in many application fields and for many different reasons. Therefore, the different users are interested in different modeling details [1]. The end user of an ADC is not interested in the conversion process or the error sources. The interesting thing is that the ADC performs as well as possible. Here, ADC models are used for calibration, i.e. increasing the accuracy of the digital representation of the analog input signal. This is done by using a model together with an error correction technique. Since the structure of the ADC is not of interest, a black-box model can be used. The ADC tester is interested in reducing the testing time. To do this, the ADC is modeled with as small a set of parameters as possible. Then only one test point per parameter is needed to fully characterize the ADC [2]. The ADC designer uses models for diagnosis, i.e. getting an understanding of error sources and why the conversion does not work as expected. This information is then used to correct design flaws. He also uses the forecasting capabilities of the model to compare the consequences of different design choices. This requires much more detailed models than for calibration or test time reduction. The system designer has needs similar to the ADC designer. However, since he is simulating an entire system, the simulation time is more critical. Therefore he does not want any unnecessary details. The important thing is that the factors that are performance limiting in the
application he is simulating are included. Also, the ADC model must be compatible with his system simulation software.

Because of the many different uses of ADC models and the fact that ADC errors are very architecture dependent [3], there is a huge amount of models presented in the literature. The majority of them only model a specific ADC architecture, but there are also many papers with general suggestions [2], [4], [5].

There is always a trade-off between good accuracy and short simulation time. Circuit-level models, such as SPICE models, may be useful to simulate small parts or, in some cases, even complete ADCs, but for systems the simulation time becomes much too large. The complexity of the systems can only be handled by using advanced CAD tools and by shifting to a higher abstraction level [6]. Behavioral modeling is necessary.

### 5.2 ADC Behavioral Modeling

A generalized model structure is suggested in [4]. It is based on the division of the converter into two main blocks: one analog and one discrete-state. There are also two blocks, A2D and D2A, which handle the conversion between analog signals and discrete states. Using this structure, various ADCs with different block diagrams can be included and thus it is possible to use the same template for different ADC architectures. Of course, to take architecture specific errors into account, models have to be developed individually for all the different architectures. To extract parameters for models of individual sub circuits, circuit level simulations are normally used. A model of a SA-ADC is shown as an example, where the analog block consists of models of the input amplifier and comparators and the D2A block of a model of the DAC.

#### 5.2.1 Static Modeling

For calibration, a black-box model is normally used, where the ADC is modeled by its transfer characteristic. The positions of all transition levels are measured, the distances from their nominal positions calculated and then a correction term can be added to each output code by the use of a look-up table. This method can be used to correct errors that exceed 1 LSB. Since a black-box model does not contain any information about the actual circuitry, it cannot be used for diagnosis.
In [3], a unified error model for integrating, successive-approximation, and flash architectures is proposed. The effects of the main error sources for each architecture are analyzed in terms of INL and DNL. Such a model can be used for calibration as well as diagnosis.

A general approach of modeling for diagnosis and calibration is the use of error signatures [2]. The behavior of an ADC is usually determined by a relatively small number of variables, such as variations in critical resistances and capacitances. If the number of variables is $x$, $x$ error signatures are used to model the ADC and the response error can be expressed as a weighted sum of these. For a certain ADC, it is then enough to investigate $x$ test points to solve the system of $x$ equations and thereby find the weights. Once the weights are known, the response error can be predicted in any test point in which the model is valid. Details on how to develop error signature models can be found in [7]. If noise is included in an error signature model, it is also possible to calculate SNDR directly, without using simulation [5].

### 5.2.2 Dynamic Modeling

For low frequencies, static ADC models may be effective but as the frequencies increase ADCs show many dynamic effects that also have to be modeled. For calibration this means that multi-dimensional correction tables have to be used. The two most common two-dimensional approaches are phase-plane compensation, where the error is modeled as a function of amplitude and slope of the input, and state-space compensation, where the error is modeled as a function of present sample amplitude and previous sample amplitude [8] [9] [10]. There are several methods to generate the error table used for correction. In [11], sine wave histograms are used to generate the error table for a model that describes the error as a function of ADC state and input slope. In [12], a dual-tone input and a bidimensional histogram is used to achieve a better error-table coverage and improved compensation. The same is accomplished in [13] with the use of pseudorandom calibration signals. A comparison is also made with an alternative compensation technique based on Volterra series. Volterra series is a mathematical approach to describe a system where nonlinear phenomena and memory effects are simultaneously present. For a given example, it is shown that the error-table approach gives better compensation and has less computational complexity. However, it is also stated that for a system with longer memory, the
Volterra approach may perform better. An advantage with the Volterra approach is that it can give some insight into system properties such as significance of various nonlinearity orders, something that is not possible with the error-table approach. Another example of a Volterra-based model can be found in [14].

For diagnosis, again, information about the individual blocks is necessary. To be useful in system design, a model also has to include the statistical properties, caused by process variation [15], that affect both static and dynamic behavior [16]. This requires advanced mathematical methods and is closely related to the ADC architecture. Some recently published models are e.g. a flash architecture in [17], a pipelined in [18], and a continuous-time sigma-delta in [19]. The models are implemented in different high-level languages, such as SIMULINK [18] and C++ [17]. A VHDL implementation of a sigma-delta model is presented in [20].

The complexity of state-of-the-art ADCs makes it difficult to develop effective dynamic models. Therefore most dynamic specifications, e.g. ENOB, do not refer to a commonly acknowledged dynamic model with which the signal response can be predicted. They serve as a description of signal degradation rather than ADC behavior and, since they are related to a specific test signal, they cannot be trusted to be the worst case for an arbitrary application [21]. They can be used to compare the performance of similar devices in similar conditions, but are of limited use for system designers who want to predict system performance [22].

5.3 References


6

An ADC Model for System Simulation

6.1 Introduction

As discussed in section 1.1, there is a great need for accurate system simulation to find reasonable ADC requirements for a certain application and as shown in section 5, there are a huge number of different ADC models. There is, however, a lack of investigations on requirements for ADC models for system simulation and their effects on system performance accuracy presented in open literature. A complex ADC model is of no use if the same information can be obtained with a simple one. An example of ADC modeling for system simulation can be found in [1], where the effect of sigma-delta converters in orthogonal frequency division multiplex (OFDM) systems is investigated.

This work has two main parts. The first is to develop a detailed ADC model that includes dynamic behavior and all performance limiting errors for a certain architecture. The model should also be validated by comparing simulated data with measurements. The second is to investigate if this accurate model is needed, if it gives any more information than simple models in system simulation. This is done by performing system simulations on realistic applications and studying how different model parameters affect system performance in comparison to simulations with a simple model.
The chosen architecture is successive-approximation. It is one of the most popular architectures and, as described in section 2.4, it has several advantages such as suitability to convert multiple signals per ADC and non-periodic multiplexed signals. The major drawback of the successive-approximation architecture is the low throughput. To obtain high sample rates, interleaving has to be used. An important advantage with choosing the successive-approximation architecture was also that there was previous experience in the research group and an ADC was available for measurements [2], [3].

The currently dominating architecture for high-speed applications is the pipelined due to its high throughput and high possible resolution. The pipelined architecture has some similarities with the SA-ADC. It also performs successive approximations only in this case the binary search is implemented in a pipeline where each stage gives a closer approximation of the input. Hence, an interleaved SA-ADC structure can be an alternative to a pipelined. Assuming that the errors introduced by interleaving can be sufficiently corrected, it should have performance comparable to a pipelined.

Since it is to be used in system simulation, the model cannot be too detailed or the simulation time will be too long. This is especially true since design normally involves the sweeping of parameters to find an optimum and thus many simulations have to be performed. As discussed in section 2.4, it is well known that the SA-ADC resolution is limited by component matching. The components that require good matching are the reference voltage generator and the comparator. Hence, it is natural to focus on finding accurate models for those. Other parts, such as the output register and the control logic for the binary search, can be assumed ideal.

One approach to increase the chance that all performance-limiting effects will be included is to keep the models as physically correct as possible. Then the model correctly mimics the behavior of the actual circuit and there is less risk that effects get lost in mathematics. A mathematical behavioral model based on explicit equations that describes the physical behavior of a circuit will be both accurate and fast. The drawback is that it will not be generally applicable.
6.2 Model Description

A model of a subranging SA-ADC has been implemented in MATLAB. As would a real ADC, it takes a time-domain analog input signal and gives a time-domain digital output signal. This makes it easy to integrate in any system simulator where embedded MATLAB code can be used. The main objective was to model all performance limiting dynamic errors as realistically as possible while keeping the model simple enough to give reasonable simulation times. This was done by using an approach with mathematical behavioral modeling as discussed in the previous section. The model equations were obtained by analytically solving the equations of an equivalent circuit found by identifying the main resistances and capacitances in the sampling frontend of a real ADC [3]. The details on this can be found in appendix B. A block diagram of the model is shown in figure 6.1. Three subranges are used: C (Coarse), M (Middle), and F (Fine). First the input is applied to get a sampled voltage $V_S$. Then, for each iteration in the binary search, a reference voltage, $V_R$, to compare to the sampled voltage is calculated by superposition of the contributions from the different subranges. Figure 6.2 shows the equivalent circuit for the contribution from the coarse subrange, $V_{R,C}$. The resistances come from the on-resistances of the NMOS switches, which are calculated as

$$r_s = \frac{1}{\mu_n C_{ox} \frac{W}{L} (V_G - V_t)}$$

where $\mu_n$ and $C_{ox}$ are process parameters, $W$ and $L$ width and length respectively of the transistor, $V_G$ is the gate-source voltage, and $V_t$ is the threshold voltage. The capacitances are the sampling capacitors. This gives a model for the settling behavior of the ADC. Errors included in the model are:

- Quantization error and clipping, modeled by a binary search among defined voltage reference levels.
- Thermal noise, modeled by random addition to the input signal.
Resistor ladder mismatch. This is modeled by a random Gaussian addition to reference voltages.

Nonlinear behavior of sampling switches. The resistance values depend on the reference values currently used.

RC settling to comparator input, modeled by analytical solution to differential equations.

Comparator recover time. Due to time constants in the comparator, some time is needed to recover from an initially wrong decision. This reduces the comparator decision time. It is modeled as a constant reduction of the available time to settle.

**Figure 6.1.** ADC model block diagram.

**Figure 6.2.** ADC model equivalent circuit.
• Jitter. This is modeled by a random Gaussian addition to the input signal phase.

Necessary model parameters are:

• Technology parameters $\mu_n$ (NMOS mobility) and $C_{ox}$ (gate oxide capacitance) to be able to calculate NMOS transistor on-resistances.

• Comparator transistor sizes to calculate $g_m$ and the parasitic capacitance of the comparator input, $C_p$ and to estimate the comparator load capacitance, $C_L$.

• Number of subranges and values of the reference generator resistors and sampling capacitors.

Since the switch on-resistances vary, the equation system has to be solved for each possible reference value, $2^n$ solutions for an $n$-bit converter. To save time at runtime, all possible roots are calculated beforehand and saved in a matrix. The calculation times for different number of bits simulated on a 1 GHz Pentium III with 512 MB RAM is shown in figure 6.3. The time increases exponentially, but this calculation has to be done only once for a model with given parameters. Calculating roots for a 12-bit ADC takes about 3 min. The simulation time vs. number of bits for a vector length of 16384 is shown in figure 6.4. The model core is very simple. It is just a binary search loop where all voltages are calculated with explicit equations. This can be implemented on a few lines of MATLAB code. The coefficients, however, are quite complex, which increases code complexity and simulation time.

6.3 Future Work

To strengthen this work, modeling of other architectures is necessary. It should be done using the same strategy of finding explicit mathematical equations based on a few physical properties of the sub blocks. Since the same blocks, T/H, reference generator (DAC), and comparator, are used
in most ADC architectures, a similar approach should be possible for other architectures as well. It remains to be seen, however, which properties of the sub blocks will be performance limiting. To model a pipelined converter would be a suitable choice since it is the dominating architecture for high-frequency applications where the dynamic effects become an issue.

Another continuation would be to use the converter model in other common applications with tough requirements on the ADC, e.g. radio receivers.

6.4 References


Part III

Papers
A MATLAB-Based ADC Model for RF System Simulations

Kalle Folkesson¹, Jan-Erik Eklund², Christer Svensson³, and Andreas Gustafsson⁴

¹,³) Electronic Devices, Department of Physics Linköping University, SE-581 83 Linköping, Sweden

²) Microelectronic Research Center, Ericsson Components AB Isafjordsgatan 16, SE-164 81 Kista, Sweden

⁴) Defence Research Establishment (FOA), Department of Microwave Technology SE-581 11, Linköping, Sweden

A MATLAB-Based ADC Model for RF System

Simulations

Kalle Folkesson\textsuperscript{1),} Jan-Erik Eklund\textsuperscript{2),} Christer Svensson\textsuperscript{3),} and Andreas Gustafsson\textsuperscript{4)}

\textsuperscript{1,3)} Electronic Devices, Department of Physics
Linköping University, SE-581 83 Linköping, Sweden

\textsuperscript{2)} Microelectronic Research Center, Ericsson Components AB
Isafjordsgatan 16, SE-164 81 Kista, Sweden

\textsuperscript{4)} Defense Research Establishment (FOA), Department of Microwave
Technology
SE-581 11, Linköping, Sweden

Abstract

Analog-to-digital converters (ADCs) are often the limiting components in signal processing systems. When a system is designed, the demands on the ADC are often set unnecessarily high to ensure that the system will work. To find reasonable demands on the ADC, an accurate model for system simulations would be useful. Such a model would also make it possible to find out which ADC errors limit the performance in a certain application and to do design trade-offs for various systems. A MATLAB-based time-domain ADC model aimed to work together with frequency domain RF simulators is presented.

7.1 Introduction

ADCs are essential parts in signal processing systems such as wireless communication and radar. Since these areas are developing rapidly, there is a need for high-performance ADCs. When a system is designed, one very important task is to choose the ADC, which often has a significant impact on the system performance. It is, however, rather difficult since there is not so much knowledge of what the requirements on the ADC
should be. This leads to ADCs with unnecessarily high performance being used, and this makes the system unnecessarily complex and expensive. One way to find suitable requirements would be to make system simulations using an accurate ADC model which includes all the performance limiting errors. That would give good knowledge about the limits in accuracy, sampling rate, and bandwidth. Such a model would also be a good tool to test error hypotheses.

This paper will focus on some of the errors that occur in ADCs and a MATLAB-based model which takes these into account will be presented.

### 7.2 Errors in ADCs

Quantization errors occur in all ADCs, even ideal ones. It arises because the analog input signal, which can take any value, is rounded to a finite number of output levels. The only way to decrease the quantization error is to increase the resolution.

Quantization is deterministic, but since the input normally consists of complicated signals and noise, the error is approximately random and has a noise-like behavior. The maximum SNR due to limitation by quantization noise for an $N$ bit ADC can for a sine wave input be calculated as:

$$\text{SNR}_{Q,max} = 20 \log \frac{V_{S,rms,\text{max}}}{V_{Q,rms}} = 20 \log \left( \frac{V_{ref}}{2\sqrt{2}} \cdot \frac{\sqrt{12}}{V_{LSB}} \right) = 20 \log \left( \frac{\sqrt{12}}{8} \cdot 2^N \right) \approx$$

$$\approx 6.02N + 1.76$$

Another important error is jitter, i.e. uncertainty in sample time. This can be considered a random error and consequently it will lead to an increase in noise level. The error amplitude

$$\Delta A \approx \frac{\partial V_S}{\partial t} \bigg|_{t=t_S} \cdot \Delta t$$

When the input signal is a sine wave

$$\Delta A = A \omega \cos \omega t \cdot \Delta t \leq A \omega \cdot \Delta t$$
Hence, if the input frequency is doubled, $\Delta A$ is doubled. That is equivalent to a 6 dB decrease in SNR. This is illustrated in figure. 7.1.

The accuracy of a system is proportional to the matching accuracy. Mismatch, i.e. gradients on a chip, will of course decrease performance. Accuracy is improved with deeper submicron technologies and it is also possible to reduce mismatch to a certain degree by using design methods that split up devices and place the parts in such a way that it minimizes the effect of gradients.

Depending on the application, mismatch can cause either noise or distortion. Therefore it is interesting to simulate for a certain application and see which effect it has in that particular case.

### 7.3 ADC Model

A MATLAB-based ADC model has been developed. To make it easy to use, it takes an analog input signal and yields a digital output. This, in addition to the fact that it is written in MATLAB, makes it compatible with evaluation software normally used to test ADCs.

ADC errors are very architecture dependent. The model simulates one of the most widespread architectures, the successive approximation ADC (SA-ADC), but it is easy to integrate models of other architectures. The model solves the differential equation that arises when the SA-ADC is modeled by an RC network. The RC time constants make it possible to model the performance decrease when the ADC runs too fast for the sampling capacitors to charge correctly. Besides the quantization error and the RC time constants, the model also takes jitter and mismatch into account.
Jitter is modeled as a normally distributed random number, which modifies the values of the sampled input signal. Since the signal is time discrete after sampling, the input frequency should not affect the SNR in the rest of the system. It is, however, clear from figure 2 that the SNR decreases significantly when the input frequency increases. This means that it is the sampling that limits the bandwidth for a certain SNR. The straight lines have a slope of -6 dB for every doubling of the input frequency. As explained in the previous section, this is the theoretical slope for a curve when jitter is the limiting factor. The simulation follows it closely for high frequencies. This means that the performance in systems with high frequency input signals will be limited by jitter. For lower frequency systems, other factors such as mismatch and thermal noise will be limiting.

Simulations like the one in figure 7.2 are useful to find out whether the performance would increase if more bits were used or if the jitter would limit it anyway.

Like jitter, static mismatch in the resistors is modeled as a normally distributed random number. Simulation results are shown in figure 7.3. As expected, SNR decreases when the mismatch increases.

The model was run together with ADS (Advanced Design System), an RF simulator from HP. A very simple test case, shown in figure 7.4, was simulated. A sine wave input signal was run without any other components, with a quantization block included in ADS, and with the model. A sample frequency of 50 MHz was used to sample a 20 MHz signal. For visibility, a 36 ps jitter was used with the ADC model. To
compare the outputs, the spectra was studied. In figure 7.5, it shows that 
the ADC model resulted in a higher noise level and harmonics.

7.4 Measurements
To verify the accuracy of the model, simulations were compared with 
measured data. This is shown in figure 7.6. The dots are measured data 
and the curve is simulated. The ADC was characterized by tuning the 
ADC parameters until the curve matched the measured data. The SNR 
measured before jitter becomes the limiting factor is unfairly low due to 
harmonics from the signal generator. When measurements were made 
using a low pass filter, the SNR for low frequencies were about 58.5 dB. 
Therefore the model was tuned to match that instead. The result was an 
ADC with 0.5% mismatch and 7 ps jitter.

An interesting phenomenon is that the jitter is higher for low 
frequencies. This may be because the sine wave clock does not produce 
very good edges then.
7.5 Conclusions

A MATLAB-based model to describe some of the most important errors that occur in ADCs was presented in this paper. It can easily run together with existing RF simulation tools to perform accurate system simulations.

Figure 7.6.
Modeling of Dynamic Errors in Algorithmic A/D Converters

Kalle Folkesson\textsuperscript{1)}, Jan-Erik Eklund\textsuperscript{2)}, Christer Svensson\textsuperscript{3)}

\textsuperscript{1,3)} Electronic Devices, Department of Physics
Linköping University, SE-581 83 Linköping, Sweden

\textsuperscript{2)} Microelectronic Research Center, Ericsson Microelectronics AB
Teknikringen 8, SE-583 30 Linköping, Sweden

Modeling of Dynamic Errors in Algorithmic A/D Converters

Kalle Folkesson¹, Jan-Erik Eklund², Christer Svensson³

¹,³) Electronic Devices, Department of Physics Linköping University, SE-581 83 Linköping, Sweden

²) Microelectronic Research Center, Ericsson Microelectronics AB Teknikringen 8, SE-583 30 Linköping, Sweden

Abstract

In communication applications, the requirements on A/D converters are high and increasing. To be able to design high-performance converters, it is important to understand the speed limitations. In this work, performance decrease caused by dynamic errors related to settling time of the switched circuits at high sampling frequencies is investigated.

8.1 Introduction

The performance requirements on analog-to-digital converters (ADCs) are continuously increased. For applications in communication, high resolution at high sampling frequencies is needed [1]. It is the aim of this work to improve the understanding of the speed limitations of algorithmic ADCs. At high frequencies, the effective resolution tends to reduce in terms of effective number of bits or signal-to-noise ratio [2]. This has two main reasons: the effect of sampling jitter and the effect of dynamic errors in the sampling and conversion processes. Such dynamic errors are for example decision time and hysteresis effects in comparators and also settling time of switched circuits. The effect of a too long comparator decision time is metastability errors, which cause spikes in the time domain [3]. In this work, dynamic effects related to settling time of the switched circuits are investigated by using a model of the successive-approximation architecture as an example. The findings are compared to measurements on an experimental ADC. The ADC is
8.2 The ADC

For the measurements, a 10-bit subranging successive-approximation ADC that uses binary search [4] was used. The principle is shown in figure 8.1. It has 3 reference voltage ranges: C (coarse), M (middle), and F (fine). The supply voltage is 5 V and the signal range is 0-2 V centered on 1 V.

The reference voltages $V_{RC}$, $V_{RM}$, and $V_{RF}$ are generated with resistor ladders that have a resistance of less than 30 $\Omega$ in series with the sampling capacitors. The sampling capacitors, $C_C$, $C_M$, and $C_F$ are 8, 2, and 1 times of a 22 fF unit capacitor respectively.

The comparator consists of several stages of differential amplifiers, clocked regenerative comparators, and latches. A simplified block diagram is shown in figure 8.2. In principle, it has two phases: Compare and Latch. The timing of these is shown in figure 8.3. When clock goes low, the reset switch opens and the evaluation cycle starts after $t_r$ ($t_r \approx 0.5$ ns), the delay in the reset switch.

Figure 8.4 shows SPICE simulations of the amplifier chain for a few input signals. To illustrate the dynamic effects, ramping input signals that

\begin{figure}
\centering
\includegraphics[width=\textwidth]{figure8.1}
\caption{Principle of the ADC.}
\end{figure}
Figure 8.2. Comparator block diagram.

Figure 8.3. Comparator timing diagram.

Figure 8.4. SPICE simulations of amplifier chain.
crosses the 0-level were chosen. The latch instant is when the reset signal goes high. When the input signal ramp crosses the 0-level at 2.5 V during the evaluation cycle, it takes some time for the amplifiers to recover from the wrong initial value, i.e. for the output signals to change from positive to negative. It shows that this recover time depends weakly on the input signal, but as a first approximation it can be considered constant.

8.3 Dynamic Errors

In this work, dynamic errors in ADCs are investigated. Primarily, an attempt is made to realistically model conversion errors due to RC time constants from resistances in reference switches and ladders together with sampling capacitors. However, another type of error is also considered; a constant time by which the comparator decision time is decreased. This will be motivated in section 8.4.

These errors limit the clock frequency at which the ADC can run. The time constants reduce the time available for nodes to settle to correct values and the constant time error reduces the comparator decision time. In both cases, a too large error leads to a wrong decision in the comparator.

8.4 The Model

A MATLAB-based model of a successive-approximation ADC has been developed [5]. The main objective was to model all limiting errors as realistically as possible. This was done by identifying the main resistances and capacitances in the real ADC and considering these an RC network to which the signals were applied. A block diagram of the model is shown in figure 8.5.

When the input signal has been applied to the RC network to get a sampled voltage, $V_S$, superposition is used to find the total contribution from the reference voltages. Figure 8.6 shows an equivalent circuit when the contribution of the coarse reference section, $V_{R,C}$, is calculated. The switches are NMOS transistors and their resistances, $r_i$, are modeled as

$$ r_i = \frac{L_S}{W_S} \frac{1}{\mu_n C_{ox}(V_{GS} - V_t)} = \frac{L_S}{W_S} \frac{1}{\mu_n C_{ox}(5 - V_{R,t} - 1.1)}, $$
where 5 is a full-scale control signal on the gate, \( V_{R,i} \) is the reference voltage, and 1.1 is an approximation of \( V_t \) when the body effect is considered. This gives resistances larger than 200 \( \Omega \). The reference ladder resistances are negligible and therefore left out of the model.

The parasitic capacitance \( C_P \) and load capacitance \( C_L \) are modeled as 0.65 pF.

Figure 8.7 shows block diagrams of two different models of the differential amplifiers and the comparator. For model \( \alpha \), the digital output is simply one or zero depending on if \( V_{out} \) is positive or negative. \( V_{out} \) is calculated as

\[
V_{out} = \frac{g_m}{C_L} \int_{t_s}^{t_k} (V_S - V_R) \, dt,
\]

where \( V_S \) is the sampled input signal, \( V_R \) the sum of all reference voltage contributions, \( V_{R,i} \), \( t_s \) is half a sampling period, and \( t_k \) is a constant time to model the recover time described in section 8.2. This reduces the time the comparator has to change an initially wrong decision. As the clock frequency, \( f_c \), increases, \( t_k \) will become a substantial part of the available decision time, \( t_k - t_s \), and performance will decrease very quickly.
Model \( b \) in figure 8.7 is an alternative model, which uses a linear model of the differential amplifiers instead of the constant delay, \( t_k \). It includes the delay in the reset switch, \( t_r \), which is set to 0.45 ns. This yields an output

\[
V_{out} = \left( \frac{g_m}{C_L} \right)^2 \int_{t_r}^{t_r} \left( V_S - V_R \right) dt\, dt
\]

Further, a model of noise on the latch input can be added by letting the digital output be random if \( |V_{out}| < V_n \). Also, hysteresis can be modeled by letting a part of \( V_{out} \) remain as an initial value when the next bit is calculated. \( V_{out,init} = K_h V_h \), where \( K_h \) is the hysteresis factor and \( V_h \) is the previous \( V_{out} \) limited to \([-2, 2]\) V. When studying the effect of one of these errors, the other is left out.

### 8.5 Measurements and Simulations

Simulations and measurements were performed with a sine wave input signal with approximately half swing. In the measurements, it was generated with a high-precision signal generator and a low-pass filter to get a signal with as little harmonic distortion as possible. The results are shown in figures 8.8 through 8.14 below. The dots are measured data and the solid lines are simulations. A slope of \(-30\) dB/octave is plotted to make it easier to compare plots. To analyze output data, a 4096-point FFT is performed and based on that the SNR is calculated. The SNR is then used as a performance measure in comparisons between measurements and simulations. The same analysis software is used to evaluate both measurements and simulations.
Figure 8.8 shows SNR as a function of clock frequency when model a is used with \( t_k = 0 \). The input signal frequency is 1.905 MHz. The performance decrease in the simulation is not nearly as large as in the measurement, so time constants alone are apparently not enough to realistically model dynamic errors.

Figure 8.9 shows the same simulation as figure 8.8 but with \( t_k = 2 \) ns. This matches the measurements very well.

To further compare model simulations and measurements, the case when \( f_c \) is approximately equal to \( f_{in} \) was studied. This gives a good view of the time domain data, see figure 8.10, but it would be much too time consuming to simulate it in SPICE. Uncertainty in the model due to inaccurate identification of ADC component values leads to that the frequencies do not match; 280 MHz was used in the measurement and 220 MHz in the simulations. It is, however, clear that the general behavior is the same.

In figures 8.11 and 8.12, the same simulations are shown with model b and \( \nu_n = 2.5e^{-1.56 \cdot 10^7 t_k} \). This models the SNR behavior very well, but fails to produce the correct form of the time domain data.

In figures 8.13 and 8.14, model b is used with \( K_h = 0.4 \). This models the time domain data correctly, but the SNR behavior is wrong.
8.5 Measurements and Simulations

Figure 8.9. SNR vs. $f_c$ with $t_k = 2$ ns.

Figure 8.10. Time domain data.

Figure 8.11. SNR vs. $f_c$ with latch input noise.

Figure 8.12. Time domain data with latch input noise.

Figure 8.13. SNR vs. $f_c$ with hysteresis.

Figure 8.14. Time domain data with hysteresis.
8.6 Conclusions
In this work, different ways to model dynamic errors in ADCs have been tested and compared to measurements. Both SNR behavior and time domain data have been investigated. It was shown that it is possible that a model that models SNR behavior correctly does not necessarily produce the correct time domain data and vice versa. This is an example of how a model can be used to verify error hypotheses.

It has also been shown that dynamic errors in successive-approximation ADCs can be modeled as a combination of time constants and a constant reduction of comparator decision time.

8.7 References
Measurement Verification of Estimation Method for Time Errors in a Time-Interleaved A/D Converter System

Jonas Elbornsson¹, Kalle Folkesson², and Jan-Erik Eklund³

¹-²) Department of Electrical Engineering Linköping University

³) Ericsson Microelectronics AB

Measurement Verification of Estimation Method for Time Errors in a Time-Interleaved A/D Converter System

Jonas Elbornsson¹, Kalle Folkesson², and Jan-Erik Eklund³

¹-²) Department of Electrical Engineering
Linköping University

³) Ericsson Microelectronics AB

Abstract
A previously presented method for estimation of time errors in time-interleaved A/D converter systems is here verified on measurements from a dual A/D converters system. The advantage of this estimation method, compared to other methods, is that it does not require any knowledge about the input signal. The estimation is most accurate for slowly varying input signals but the signal quality is improved even when the estimation is done for a sinusoidal signal close to the Nyquist frequency.

9.1 Introduction
Many digital signal processing applications, such as radio base stations or VDSL modems, require ADCs with very high sample rate and very high accuracy. To achieve high enough sample rates, an array of \( M \) ADCs, interleaved in time, can be used. Each ADC should work at \( 1/M \)th of the desired sample rate [1], [2], see figure 9.1. Three kinds of mismatch errors are introduced by the interleaved structure:

- Time errors
- Offset difference
- Gain difference
We consider only the time errors in this paper. The time errors are assumed to be static, so that the error is the same in the same ADC from one cycle to the next.

Methods for estimation of timing errors have been presented in for instance [3] and [4] but those methods require a known calibration signal. Calibration of ADCs is time-consuming and expensive. Therefore a lot of costs can be saved if the errors in the ADC can be automatically estimated and compensated for at run-time.

We will in this paper review an estimation method for time errors in interleaved ADCs, [5], [6]. The estimation method does not require any prior knowledge about the input signal, except that it should be band limited to the Nyquist frequency. The estimated time errors are then used for correcting the output signal. In order to show the quality of the estimates, we correct the data by interpolation in the frequency domain. The results in [5], [6] are based on simulations. The estimation method is here verified on measurements from a dual A/D converters system.

9.2 Theory

In this section the estimation and compensation algorithms are briefly described. A more complete description of the estimation algorithm is given in [5], [6].
9.2.1 Notation

The analog input signal is denoted \( u(t) \). \( T_s \) denotes the nominal sampling interval. \( M \) is the number of ADCs. The time error for the \( i \)th ADC is denoted \( t_i \). The output from the \( i \)th ADC is denoted \( y_i[k] \) where \( k \) is the \( k \)th sample from that ADC. \( N \) denotes the number of samples from each ADC. \( y[k] \) denotes the non-uniformly sampled signal and \( \tilde{y}^0[k] \) denotes the estimated uniformly sampled signal.

9.2.2 Time error estimation method

The basic idea of the estimation method is that the signal changes more on average if the sampling interval is longer than the nominal sampling interval and vice versa, see figure 9.2. We assume, for this estimation algorithm, that the input signal is band limited.

9.2.2.1 Estimation algorithm

A crude estimate of the time errors is first calculated as

\[
t_i^{(0)} = T_s \sum_{j=2}^{i} \left( \sqrt{\frac{\hat{R}^N_{j,j-1}[0]}{M \sum_{i=1}^{M} \hat{R}^N_{r,i-1}[0]}} - 1 \right)
\]

(1)

\( i = 2, \ldots, M \)
where \( \hat{R}_{i,i-1}[0] \) is calculated from measured data as

\[
\hat{R}_{i,i-1}[0] = \frac{1}{N} \sum_{k=1}^{N} \{y_i[k] - y_{i-1}[k]\}^2 \tag{2}
\]

The time error estimates can then be improved by fixed-point iteration:

\[
t_i^{(l)} = T_s \sum_{j=2}^{i} \left( \sqrt{\frac{\hat{R}_{j,j-1}[0]}{a^{(l-1)}} - 1} \right)
\]

\[
a^{(l)} = M \left( 1 + \frac{2}{M} \sum_{i=1}^{M} \left( \frac{t_i^{(l)}}{T_s} \right) - \frac{2}{M} \sum_{i=1}^{M} \frac{t_i^{(l)}}{T_s} \right)
\]

### 9.2.3 Correction Through interpolation

When the time errors are estimated, we need to estimate the uniformly sampled signal from the measured non-uniformly sampled signal. The reconstruction is done in the frequency domain \([7]\). Calculate the DFTs of the \(M\) subsequences \(y_i[k] \), \( i = 1, \ldots, M \):

\[
Y_i[n] = \text{DFT}\{y_i[k]\} \tag{4}
\]

The DFT of \( \hat{y}_i^0[k] \) can then be calculated from \( Y_i[n] \) as

\[
\hat{Y}_i^0[n] = e^{-j\frac{2\pi n t_i}{MN}} Y_i[n], \quad n = -N/2, \ldots, N/2 - 1 \tag{5}
\]

\( \hat{Y}_i^0[n] \) can then be calculated from these \(M\) subsequences \([8]\)

\[
\hat{Y}^0[n] = \sum_{i=1}^{M} e^{-j\frac{2\pi (i-1)n}{MN}} \hat{Y}_i^0[(n \text{ mod } N) - N/2] \tag{6}
\]

\( n = -NM/2, \ldots, NM/2 - 1 \)
The estimated uniformly sampled signal is then calculated as

$$\hat{y}^0[k] = \text{IDFT}\{\hat{Y}^0[n]\}$$

(7)

### 9.2.4 Time Error Distortion

The frequency synthesizer that is used as signal source typically has high harmonic distortion. However, the position of the distortion caused by the time error is, for a dual ADCs system, given by $f_N - f_{in}$, where $f_N$ is the Nyquist frequency and $f_{in}$ is the input signal frequency. This means that we can study only the improvement of the tone caused by the time error without having to bother about the quality of the signal source. The distance between the energy of the signal peak and the energy of the distortion peak caused by the time error is measured, see figure 9.3. We will denote this measure signal-to-time-distortion ratio (STDR). We will here calculate the signal quality as a function of the size of the time error for a sinusoidal input, $u(t) = \sin(\omega t)$. We assume, for simplicity in the calculations, that $M = 2$ and that $\omega = \frac{2\pi a}{NM}$ where $a$ is an integer. This means that $y[k]$ is formed from the two subsequences

$$y_1[k] = \sin\left(\frac{2\pi a}{NM}2k\right)$$

$$y_2[k] = \sin\left(\frac{2\pi a}{NM}(2k + 1 + t_2)\right)$$

**Figure 9.3.** Definition of the Signal-to-Time-Distortion Ratio.
The DFT of \( y[k] \) is

\[
Y[n] = \begin{cases} 
\frac{N}{2} (1 + e^{j \frac{\pi n}{N} t_2}) & \text{if } n = a \\
\frac{N}{2j} (-1 + e^{-j \frac{\pi (N-a)}{N} t_2}) & \text{if } n = N - a \\
\frac{N}{2} (-1 + e^{-j \frac{\pi n}{N} t_2}) & \text{if } n = N + a \\
\frac{N}{2j} (1 + e^{j \frac{\pi (2N-a)}{N} t_2}) & \text{if } n = 2N - a \\
0 & \text{otherwise}
\end{cases}
\] (8)

The signal-to-time-distortion ratio is then

\[
STDR = 20 \log \left( \left| \frac{1 + e^{j \frac{\pi n}{N} t_2}}{1 - e^{-j \frac{\pi n}{N} t_2}} \right| \right)
\] (9)

A Taylor expansion of (9) gives

\[
STDR \approx 20 \log 2 - 20 \log \left( \frac{2 \pi at_2}{2N} \right)
\] (10)

This means that the signal-to-time-distortion ratio is decreased approximately 6 dB per octave, see figure 9.6

**9.3 Measurements**

In this section, the measurement verification of the algorithm described in section 9.2, is presented.

**9.3.1 Measurement setup**

Measurements were done using two AD6644 evaluation boards from Analog Devices [9], with a sampling frequency of 66.6 MHz each, see figure 9.4(a). A signal generator was used as clock signal and a differential pulse splitter was used to create two clock signals with opposite phase, thereby doubling the sampling frequency. A signal generator was used as input signal, see figure 9.4(b).
9.3.2 Data acquisition

The measured data were collected from the logic analyzer and MATLAB was used for signal processing. To look only at time errors, offset and

---

**Figure 9.4.** (a) Two AD6644 evaluation boards. (b) Measurement setup: Clock signal and input signal are generated from the two signal generators. Time interleaving is achieved by inverting the clock signal to one ADC. The output signal is collected in the logic analyzer and the signal processing is done in MATLAB.
9.3 Measurements

Gain errors need to be eliminated. This can be done by various mean value calculations and is not a subject of this paper.

The time error estimation algorithm was evaluated with sinusoidal input signals. Nine batches of data at different frequencies between 5 MHz and 60 MHz were collected. For each input signal frequency, 65536 samples per channel were collected.

9.3.3 Evaluation

The estimation algorithm does not utilize any prior information about the input signal. Therefore, the fact that we know that the input signal is sinusoidal is not used in the algorithm and does not influence the performance of the estimation algorithm. The reason why we have chosen a sinusoidal input signal is that it is easy to generate and that there exist good signal quality measures for sinusoidal signals. We use the signal-to-time-distortion ratio for evaluation of the signal quality, see section 9.2.4.

9.3.3.1 With extra delay for visibility

In order to visualize the effect of the time error, the clock to one of the ADCs was delayed 3.2 ns \( \approx 0.4T_s \) by adding a 0.5 m coaxial cable. Figure 9.5 shows the measured signal with and without compensation for the timing error. The compensation was here done by moving the
sampling instances with the estimated time error. The input signal frequency was here 10 MHz.

9.3.3.2 Normal operation

The time error is here much smaller, 0.17 ns ≈ 0.02 $T_s$, and can not be seen in the time domain. Instead we study the distortion component in the frequency domain before and after estimation and compensation as described in section 9.2. Figure 9.6 shows the theoretical and measured signal-to-time-distortion ratio without time error compensation. Figure 9.7(a) shows the improvement of the signal-to-time-distortion ratio after compensation and in figure 9.7(b) the signal-to-time-distortion ratio after compensation is shown. The result of the estimation is here shown for three input signals:

- A 10 MHz sinusoidal signal, 131072 samples.
- A 40 MHz sinusoidal signal, 131072 samples.
- Sinusoids of nine different frequencies between 5 MHz and 60 MHz concatenated into one signal, 1179648 samples.

9.4 Conclusions

We have evaluated an estimation method for time errors on measurements from a time-interleaved A/D converter system. The method does not require any knowledge about the input signal. We have done the evaluation with sinusoidal input signals at different frequencies. The knowledge about the input signal is not used in the estimation algorithm. The choice of input signal is motivated by the fact that it is easy to generate sinusoidal signals and that there exist good signal quality measures for them.

We have verified, with many different input signals, that the signal quality is improved after compensation. The measurements show that the improvement is better for lower frequencies and tends to zero near the Nyquist frequency. This result agrees with the theory [5], [6].
9.4 Conclusions

![Graph showing comparison between theoretical and measured signal-to-time-distortion ratios without compensation.](image)

**Figure 9.6.** Comparison between theoretical Signal-to-Time-Distortion Ratio calculated from Equation (9) (dashed line) and measured Signal-to-Time-Distortion Ratio (solid line), without compensation.

![Graph showing improvement of signal-to-time-distortion ratio after error compensation.](image)

**Figure 9.7.** (a) Improvement of signal-to-time-distortion ratio after error compensation. (b) Signal-to-time-distortion ratio after time error compensation. The time error estimation has been done for different input signals: Sinusoidal signal at 10 MHz (solid), Sinusoidal signal at 40 MHz (dashed), and Sinusoidal signals between 5 MHz and 60 MHz concatenated (dash-dotted). The uncompensated curve is shown as reference (dotted).
9.5 References


Relevance of Using Single-Tone Tests to Characterize ADCs for ADSL Modems

Kalle Folkesson\textsuperscript{1), Jan-Erik Eklund\textsuperscript{2)}, Christer Svensson\textsuperscript{3)}

\textsuperscript{1,3)} Electronic Devices, Department of Electrical Engineering Linköping University, SE-581 83 Linköping, Sweden

\textsuperscript{2)} Microelectronic Research Center, Ericsson Microelectronics AB Datalinjen 3, SE-581 15 Linköping, Sweden

Relevance of Using Single-Tone Tests to Characterize ADCs for ADSL Modems

Kalle Folkesson\textsuperscript{1)}, Jan-Erik Eklund\textsuperscript{2)}, Christer Svensson\textsuperscript{3)}

\textsuperscript{1,3)} Electronic Devices, Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

\textsuperscript{2)} Microelectronic Research Center, Ericsson Microelectronics AB
Datalinjen 3, SE-581 15 Linköping, Sweden

Abstract
The ADC in ADSL modems is chosen to meet a requirement of an effective number of bits. To find the effective number of bits of an ADC, a single-tone test is used. Since an ADSL application is very different from the single-tone, it is not certain that this is suitable method. A detailed ADC model has been included in a MATLAB model of an ADSL link and simulations have been made to investigate the relevance of single-tone characterization of ADCs for ADSL applications.

10.1 Introduction
Asymmetric digital subscriber line (ADSL) technology is designed to meet the need for high-speed Internet connections for homes and small offices. It can provide up to 8 Mbit/s downstream and up to 1 Mbit/s upstream using the existing twisted pair (TP) copper cables. An ADSL link consists of one ADSL transceiver unit on the provider side (ATU-C) and one on the customer side (ATU-R), also known as an ADSL modem. The frequency range used is 4 kHz to 1.104 MHz. This is divided into 256 subchannels with a discrete multi tone (DMT) modulation using subcarriers with a frequency spacing of 4.3125 kHz. The subcarriers are individually QAM modulated with 2 to 15 bits depending on the signal-to-noise-and-distortion ratio (SNDR) of each channel. The higher the SNDR in a channel, the more bits will be assigned to that particular channel. With the frequency division multiplex (FDM) mode of
operation, which is used throughout this work, the first 32 channels (4-
138 kHz) are used for upstream data and the rest are used for downstream
data. The frequency range up to 4 kHz is reserved for plain old telephone
service (POTS).

The analog-to-digital converter (ADC) in an ADSL modem is
normally chosen to meet an SNDR requirement [1]. This requirement is
translated to effective number of bits (ENOB) by single-tone
characterization. Since the ADSL application is a multi-tone system, the
signals, however, are very different from the single-tone case. In this
work, simulations are made to investigate if the performance in terms of
ENOB found from a single-tone characterization is relevant for ADSL
systems. This is done by comparing simulations where different error
mechanisms are active in the ADC. In section 10.2, the models used are
described and the simulations can be found in section 10.3 followed by
some conclusions in section 10.4.

10.2 Model Descriptions

10.2.1 ADSL Model
A MATLAB model of an ADSL link [2] was expanded with a channel
model for the downstream case and an accurate ADC model [3], [4],
which is described more closely in the next section. A block diagram of
the downlink is shown in figure 10.1. The uplink looks the same with the
exception of the digital filters, which then are low-pass. Since it is the
impact of ADC errors on system performance that is being investigated,
the digital-to-analog converters (DACs) and all filters are ideal. The
transmitter programmable gain controls (PGCs) set the levels of the
transmitted signals to -38 dBm/Hz for the uplink and -40 dBm/Hz for the
downlink according to the standard [5]. To model the TP cable, a number
of standard test loop models and parameters are available [2], [6]. The
purpose of the receiver PGC is to optimize the dynamic range of the
ADC. If the input voltage is too low, some levels will never be used,
which reduces the ENOB. If, on the other hand, the input voltage is too
high, clipping will occur. To meet the bit error rate (BER) requirement of
10^-7 [5], too much clipping in the ADC is not allowed. Since the DMT
signal has a large number of carriers, it can be considered to have a
Gaussian distribution. To meet the bit error rate requirement, a headroom
of at least 5.6 times the standard deviation of the DMT signal, \( \sigma_{DMT} \), is
necessary [7]. To have some extra margin, the maximum level is chosen as $6\sigma_{\text{DMT}}$. After A/D conversion and filtering, the signal is equalized first in the time domain (TEQ) and then in the frequency domain (FEQ). The FFT and IFFT use 512 bins and the model sampling rate is 2.208 MS/s.

A block diagram describing the noise taken into account is shown in figure 10.2. The noise, $v_n$, is the thermal noise of the TP cable and has a level of -140 dBm/Hz [5]. When the transmitter is transmitting, the receiver on the same side experiences an echo of the data being sent. This echo, $v_e$, is simply modeled as the transmitted signal attenuated 14 dB. Crosstalk induced on the receiver by transmitters on other TP cables is not taken into account.

To measure the link performance, a known set of random 4-QAM modulated symbols are sent on all channels. The SNDR for each channel can then be calculated by comparing the sent data with what is received. The SNDR for a channel is used to calculate the bit load for that channel, i.e. assigning an appropriate number of bits to it. In this work, the sum of all bit loads, the total bit load, is used as a performance measure. The ADSL symbol rate is 4 kHz, which gives a bit rate of 4000 times the total bit load.
10.2 Model Descriptions

10.2.2 ADC Model

The ADC is a subranging successive-approximation ADC that uses binary search [3], [8]. Four time-interleaved cells are used. A principal block diagram for one cell is shown in Figure 10.3. Three subranges are used: C (Coarse), M (Middle), and F (Fine). The input signal is sampled and then, for each iteration in the binary search, resistor ladders are used to generate a reference voltage, $V_R$, to compare to the sampled voltage, $V_S$.

Errors in ADCs are very architecture dependent. In this work, three different error mechanisms are investigated. The first is static mismatch in the resistor ladders, which is modeled as a Gaussian random number added to the resistance values. The error is varied by changing the standard deviation of that number. The resistor value used in the simulation is calculated as

$$ R = R(1 + \alpha) , $$

Figure 10.3. Principal ADC block diagram.
where \( x \) is a Gaussian random number with the standard deviation \( \sigma_x = 1 \). The error thus has a standard deviation of \( R\sigma \). The error varied in the simulations is the relative standard deviation, \( R\sigma \). The second error mechanism is static mismatch in the capacitors, which is modeled the same way as the resistor mismatch. The third is settling errors. Settling errors are dynamic errors that occur due to time constants in the RC link made up by the resistance in resistor ladders and switches and the capacitance in sampling capacitors. These time constants set a lower limit for how much time is needed to charge the capacitors. The time constants in this case are in the range of about 0.3 to 2.5 ns. When there is not enough time available to charge capacitors to the correct values, errors will occur. The available time for settling is calculated as

\[
t = \frac{1}{2f_c} - t_k,
\]

where \( f_c \) is the ADC clock frequency and \( t_k \) is a constant time loss in the comparator. The error is varied by changing \( t_k \), which has been shown to be important for modeling of dynamic errors in successive-approximation ADCs [4]. Hence, there are two different mechanisms taken into account: one related to the clock frequency, which is fix, and one related to the time loss in the comparator, which is the error that is varied in the simulations. The effect of the clock frequency related error is negligible in the studied test cases.

**10.3 Simulations**

**10.3.1 Simulation Setup**

A run of the model has several states. A test signal is sent while the PGCs are set and the TEQ and FEQ coefficients are calculated. After that, random data is sent and the performance measured. First, the uplink is run and the signal saved when random data is transmitted. This signal is then used as echo when the downlink is run. The model is run for different values of the ADC errors. For each test case, three simulations are run; one for each error mechanism. Only one error mechanism is included at a time.
10.3 Simulations

Test case 1 is a 3-km loop of the type ETSI test loop 2 [6]. Channels 6-31 are used for the upstream data and channels 35-254 for downstream data. The ADC is clocked at 35.328 MHz. Since each sample takes 16 cycles to produce and there are 4 parallel ADC cells, this gives a sampling frequency of 8.832 MS/s. For the performance measurement, 600 random symbols are used. The relative standard deviations of resistor ladders and capacitors are varied from 0 to 10% and the comparator time loss is varied from 0 to 13 ns. Since the available settling time without time loss is approximately 14 ns, the upper limit gives a very large error. In test case 2, a shorter loop is used. The parameters are the same as in test case 1, with the exception of the loop length, which is now set to 1 km. Test case 3 is the same as test case 1, but without any echo included.

The ADC is also simulated in single-tone tests. A sine wave is used as input and the ENOB is calculated by measuring the SNDR of the output. To avoid clipping, the input amplitude is set to 90% of full swing. A random noise 10 dB below the quantization noise is added. Using this method, the ENOB as a function of the error amplitude is determined for each error mechanism. This means that an ENOB in the ADC is translated to an equivalent error value. To investigate if the type of error affects the total bit load for a certain effective number of ADC bits, the total bit load is plotted as a function of the ENOB. For a certain ENOB in the single-tone simulation, the corresponding error values of the different error mechanisms are found. Then the total bit loads that correspond to those error values are found in the ADSL simulation.

10.3.2 Simulation Results

The results from the single-tone tests are shown in figure 10.4. Here it can be seen how many ENOB a certain error value corresponds to. Each simulation point represents an outcome of a statistic process. The stars are a moving average over 9 points, which is included to illustrate the trend.

Figure 10.5 shows the bit load for each channel. The performance is not very good, but the filters have not been optimized and there is no special echo cancellation included. Since the bit load of a certain channel is directly related to the SNDR of that channel, it can be seen that high frequencies have a lower SNDR than low frequencies. This is because high-frequency signals are attenuated more in the TP cable. Increasing the value of an ADC error is equivalent to increasing the noise floor. This
will decrease the SNDR of all channels and as the noise floor keeps increasing, more and more channels will be lost completely starting with the high-frequency ones.

The ADSL performance when changing the different ADC errors for test case 1 is shown in figure 10.6. As before, the stars are a moving average over 9 simulation points. This plot gives a relation between the size of an error and the total bit load. It can be seen that the curve for error in capacitors varies a lot more than the one for error in resistor ladders. This is because the capacitors affect all values and since there are so few of them, different values of the random variable will cause quite large variations. Since not all parts of the resistor ladders are connected all the time, an error there only affect values in a certain interval. Another thing worth noticing is the very steep slope of the curve for settling errors. As $t_k$ increases, it removes a substantial part of the available settling time and the performance decreases very quickly [4].

By finding the resistor ladder error corresponding to a certain ENOB from the single-tone test in figure 10.4 and then finding the total bit load corresponding to that resistor ladder error from the ADSL simulation in figure 10.6, the total bit load is plotted as a function of the ENOB in figure 10.7. By using this method, the relation between ENOB and total bit load is found. The same simulation has also been done for the other error mechanisms and for all three test cases. The result can be seen in figure 10.8. The ADSL performance is of course better for the cases with a shorter loop and without the echo. It can be concluded that for a certain ENOB from a single-tone test, the performance of the ADSL modem in terms of total bit load varies depending on which error mechanism is

---

**Figure 10.4.** ADC performance in single-tone test.  
**Figure 10.5.** Bitload per channel.
active in the ADC. If, for example, points A and B are compared, it can be seen that the total bit load differs 430 bits, which corresponds to a bit rate difference of over 1.7 Mbit/s. If points A and C are compared, it shows that the ADC resolution differs 2 bits for the same total bit load.

10.4 Conclusions

A detailed ADC model was introduced into a MATLAB model of a full ADSL link. This model was used for estimating total ADSL bit load as a function of ADC quality and comparing this to a simple ENOB characterization of the ADC. It was found that the capacity of the ADSL link can vary over 1.7 Mbit/s for a certain ENOB, depending on which error mechanism is active in the ADC. Equivalently, the ENOB can vary.
up to 2 bits for a given ADSL capacity. A simple ENOB number is thus not sufficient as ADC characterization. Instead, a detailed ADC model is recommended in communication link modeling.

10.5 References

An Accurate ADC Model in Radar System

Simulation

Kalle Folkesson and Christer Svensson

Electronic Devices, Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

to be presented at the International Workshop on ADC Modelling and Testing, Sep. 2003
An Accurate ADC Model in Radar System

Simulation

Kalle Folkesson and Christer Svensson

Electronic Devices, Department of Electrical Engineering
Linköping University, SE-581 83 Linköping, Sweden

Abstract
To show the usefulness of performing system simulations with an accurate ADC model, some simulations of a radar receiver are presented. The accurate model gives very different results than simple characterization with SNR and SFDR. System performance varies quite much for ADCs with the same specified performance in terms of SNR or SFDR depending on which error mechanism is active in the ADC. With this detailed knowledge of how the ADC affects system performance, the ADC requirement margin can be reduced thus saving cost and power consumption.

11.1 Introduction
Modern design methods are based on early system modeling and simulation. Analog-to-digital converters (ADCs) are normally described as a simple quantizer with a given number of bits (NOB), which translates to a certain signal-to-noise ratio (SNR). However, ADCs normally have more errors than the simple quantization error so a more accurate modeling may be useful [1], [2], [3]. An important outcome from simulations with a more accurate model is that the performance requirement margin of the ADC can be minimized, thus saving cost and power consumption. Accurate models have previously been shown to be useful for DSL applications [4].
In this work, the usefulness of an accurate ADC model in system simulation is investigated by studying the effects of various ADC errors on the system performance of a radar receiver. A comparison is made with simple ADC characterization with SNR and spurious-free dynamic
range (SFDR). An accurate ADC model, written in embedded MATLAB code [5], was included in Agilent ADS system simulator. A complete Frequency Modulated Continuous Wave (FMCW) radar receiver chain including RF frontend, ADC, digital filters, and post processing was implemented in the system simulator. The models are described in section 11.2 and the simulation results presented in section 11.3 followed by conclusions in section 11.4.

11.2 Model Descriptions

11.2.1 Radar Model
In an FMCW radar, phase and Doppler shifts are used to find the range to and speed of a target [6]. Depending on the range to the target, the input signal to the radar receiver can vary very much in power. A detection threshold is set and any signal above that is recognized as a target. Hence, all harmonics have to be lower than the detection threshold or false targets will appear. Possible multiple input signals with large power differences set tough requirements on the radar receiver. To be able to detect weak targets, the detection threshold has to be set low, which requires spurious peaks originating from strong targets to be even lower. Apart from harmonic distortion, strong targets will also mix with other signals and harmonics creating strong spurious peaks. A relevant measure of system performance for this type of radar is the amplitude difference between a weak target and the largest spurious peak. This can be defined as system level SFDR. It is then assumed that the noise level always is lower than the spurious peaks, which was true for all the cases studied in this work.

The radar receiver is modeled in three blocks as shown in figure 11.1. The RF block contains predefined ADS components such as filters, amplifiers, and a mixer as shown in figure 11.2. There are a number of advanced parameters to specify the components, but it can also be made fairly simple by using default values. Filters are characterized with cut-off frequencies, pass band gain, and stop band attenuation, the amplifiers with gain and noise figure, and the mixer with LO and RF rejections, conversion gain, and noise figure. The ADC and digital filters blocks were implemented in MATLAB. MATLAB was also used to generate input signals and for processing output signals. The ADC model is described in the next section.
11.2.2 ADC Model

A MATLAB-based time-domain model of a subranging successive-approximation ADC that uses binary search has been developed [5], [7]. The main objective was to model all performance limiting errors as realistically as possible. This was done by identifying the main resistances and capacitances in the sampling frontend of a real ADC [8] and considering these an RC network to which the signals are applied. A block diagram of the model is shown in figure 11.3. Three subranges are used: C (Coarse), M (Middle), and F (Fine). The input is sampled and then, for each iteration in the binary search, resistor ladders are used to generate a reference voltage, $V_R$, to compare to the sampled voltage, $V_S$. Figure 11.4 shows the equivalent circuit for the contribution from the coarse subrange, $V_{R,C}$. The resistances come from the on-resistance of the MOS switches and the capacitances from the sampling capacitors.

In the simulations, the effect of three different error mechanisms will be investigated. The first is static mismatch in the resistor ladders, which causes spread in the generated reference voltages. It is modeled as a Gaussian random number added to the resistance values as

$$R = R_0 (1 + \sigma_R x),$$

\[
\text{Figure 11.1.} \text{ Radar model block diagram.}
\]

\[
\text{Figure 11.2.} \text{ Radar receiver RF part.}
\]
where $R_0$ is the nominal resistance and $x$ a Gaussian random number with the standard deviation $\sigma_x = 1$. The error is varied by changing the relative standard deviation of the resistance, $R_0\sigma_R$. The second error is comparator recover time, i.e. the time it takes for the comparator to change an initially wrong decision. This is modeled as a constant time loss that limits the time available for settling. The available time for settling is calculated as

$$t = \frac{1}{2f_c} - t_k,$$

Figure 11.3. ADC model block diagram.

Figure 11.4. ADC model equivalent RC network.
where \( f_c \) is the clock frequency and \( t_k \) is the constant time loss. The error is varied by changing \( t_k \), which has shown to be an important parameter for modeling dynamic errors in SA-ADCs [5]. The third error is NOB. Then, the simulations are equivalent to using a simple ADC model, which only includes ideal quantization and clipping.

### 11.3 Simulations

#### 11.3.1 Simulation Setup

The radar uses a 10 GHz RF frequency, which is mixed down to an IF of 360 MHz in the receiver. This is then sampled with 160 MHz. To fulfill the ADC requirements, 16 parallel 12-bit cells, each with a sampling frequency of 10 MHz are used, giving a total sampling frequency of 160 MHz. The sweep bandwidth is 150 MHz, the modulation time 3.2 \( \mu \)m, and the number of repeated pulses 40. This gives a range resolution of 1 m and a speed resolution of 117 m/s. The speed resolution is quite low, since increasing it means a significantly longer simulation time.

The test case is a short-range missile detection system. It includes two targets. Target 1, the target of interest, is a weak target at 19 m range moving at a speed of 798 m/s and gives an input signal power of -35 dBm. Target 2 is a strong interfering target at 42 m range moving at 1178 m/s and giving the input signal power -15 dBm. When measuring the system performance, target 2 is removed completely and radar SFDR is measured from the peak of target 1 to the highest spurious peak.

#### 11.3.2 Simulation Results

A two-dimensional plot of the radar image is shown in figure 11.5. It is obtained using a standard method that, among other things, involves performing FFTs on the output signal. Since, however, all frequencies used in the FFTs are not used in the system, it is only relevant to look for targets in the lower part of the figure.

To investigate how a certain ADC error affects the system performance, a number of simulations were run for different values of the different errors. Figure 11.6 shows how radar performance varies as a function of the different errors. Since the curve for resistor ladder mismatch varies quite much depending on the outcome of the statistic variable, the trend is also shown with stars representing a moving
average over 7 points. Simulations like these are very useful to find suitable requirements for an ADC in a system.

The ADC is also simulated in single-tone tests to represent a typical characterization. The results are shown in figure 11.7. A sine wave is used as an input and the SNR is calculated by performing an FFT on the output. To avoid clipping, the input amplitude is set to 90% of full swing. With these tests, the ADC SNR as a function of error amplitude is determined for each error mechanism.

To investigate how the error type affects the radar performance for a certain ADC SNR, the radar SFDR is plotted as a function ADC SNR in figure 11.8. This plot is obtained by finding the equivalent error amplitude for a certain ADC SNR from the single-tone test and combining it with the radar SFDR that corresponds to that error amplitude in the radar simulation. Again, for visibility, the mismatch curve is presented as a moving average over 7 points. It is clear that ADC SNR is not a suitable performance measure, since radar performance differs >10 dB depending on error type. Equivalently, the requirement on ADC SNR can differ >20 dB for a certain radar SFDR.

Figure 11.9 shows the same simulations as figure 8, but with SFDR used as a measure of ADC performance instead of SNR. Since radar SFDR is the performance limiting factor of the system, ADC SFDR is, of course, a better measure than SNR, but even here system performance and ADC requirement can differ up to 6 and 10 dB respectively depending on error type.
11.4 Conclusions

A successful combination of an accurate ADC model and a radar receiver system implemented in Agilent ADS simulation framework has been demonstrated. It has also been shown how this model can be used to predict the effects of various ADC errors on the radar system performance.

For the actual radar system, where system SFDR is the limiting factor, it was shown that ADC SNR, or the corresponding NOB, is not a suitable measure of ADC performance. The requirement of the ADC for a given system performance can vary up to 20 dB depending on which error mechanism dominates in the ADC. As expected, ADC SFDR is a better
measure, but still the ADC requirement can differ up to 10 dB. For these reasons, an accurate ADC model of the type presented here should replace the simple quantizer model in radar system simulation.

11.5 References


Part IV

Appendix
A.1 SNR

Assume an ADC that performs an $n$-bit quantization of a range $V_{\text{ref}}$. This gives a quantization step of

$$V_{\text{LSB}} = \frac{V_{\text{ref}}}{2^n}.$$ 

The quantized signal is shown in figure A.1 and the error in figure A.2.

**Figure A.1.** Jitter. **Figure A.2.** Quantization error.
The quantization noise will limit the SNR performance to

\[ \text{SNR} = 10 \log \left( \frac{P_s}{P_n} \right) = 20 \log \frac{V_{S,\text{RMS}}}{V_{Q,\text{RMS}}} \ [\text{dB}] . \]

For a full-scale sine wave input

\[ V_{S,\text{RMS}} = \left( \frac{1}{T} \int_{-T/2}^{T/2} \left( \frac{V_{\text{ref}}}{2} \sin^2 (t) \right) dt \right)^{1/2} = \frac{V_{\text{ref}}}{2} \left( \frac{1}{T} \int_{-T/2}^{T/2} \frac{1}{2} - \frac{\cos(2t)}{2} \ dt \right)^{1/2} = \]

\[ = \frac{V_{\text{ref}}}{2T} \left( \frac{t}{2} - \frac{\sin(2t)}{4} \right)_{-T/2}^{T/2} = \frac{V_{\text{ref}}}{2\sqrt{2}} = \frac{V_{\text{LSB}} 2^n}{2\sqrt{2}} . \]

The quantization error

\[ V_{Q,\text{RMS}} = \left( \frac{1}{T} \int_{-T/2}^{T/2} V_Q^2 dt \right)^{1/2} = \left( \frac{1}{T} \int_{-T/2}^{T/2} \left( -\frac{V_{\text{LSB}} \cdot t}{T} \right)^2 dt \right)^{1/2} = \]

\[ = \left( \frac{V_{\text{LSB}}^2}{T^3} \left[ \frac{t^3}{3} \right]_{-T/2}^{T/2} \right)^{1/2} = \frac{V_{\text{LSB}}}{\sqrt{12}} . \]

and the SNR becomes

\[ \text{SNR} = 20 \log \left( \frac{V_{\text{LSB}} 2^n \cdot \sqrt{12}}{2\sqrt{2}} \right) = 20 \log \left( \frac{3}{2} 2^n \right) \approx 6.02n + 1.76 \ [\text{dB}] . \]
A.2 Jitter

Figure A.3 illustrates how the jitter-induced error doubles when the input frequency doubles for a certain jitter. The SNR is affected as

\[
\text{SNR}_2 - \text{SNR}_1 = 20 \log \frac{V_S}{2V_N} - 20 \log \frac{V_S}{V_N} = 20 \log \frac{1}{2} = -6 \text{ dB},
\]

where \(V_S\) is the RMS signal power and \(V_N\) and \(2V_N\) are the RMS voltages of the jitter-induced noise for the two cases. Hence, jitter will degrade the SNR by 6 dB/oct.

\[\text{Figure A.3. Jitter.}\]
Model Equations

In this section, the model equations for the SA-ADC shown in figure B.1 are derived. $V_{Ri}$ are reference voltages, $S_i$ switches, and $C_i$ sampling capacitors. $C_P$ is the parasitic input capacitance of the comparator and $C_L$ the load. There are three things that must be calculated. First the settled reference and sampled voltages of the comparator input, $V_R$ and $V_S$, have to be found. Then these are used to find the comparator output, $V_{out}$.

Figure B.1. ADC block diagram.
B.1 Calculate Settled Reference Voltage

An equivalent representation of the sampling frontend when generating reference voltages is shown in figure B.2.

\[
V_{\text{ref}} = H \cdot V_{\text{ref}} = \frac{Z}{1 + sr_1 C_1} V_{\text{ref}} = \left\{ \begin{array}{l}
Z_{\text{num}} = \text{numerator of } Z \\
Z_{\text{den}} = \text{denominator of } Z
\end{array} \right. =
\]

\[
v_R = \frac{sC_1 Z_{\text{num}}}{(1 + sr_1 C_1)Z_{\text{den}} + sC_1 Z_{\text{num}}} V_{\text{ref}}
\]

\[
v_R \left[ (1 + sr_1 C_1)Z_{\text{den}} + sC_1 Z_{\text{num}} \right] = sC_1 Z_{\text{num}} V_{\text{ref}}
\]

\[
Z = \sum_{i=2}^{n} \frac{1}{sC_i} \prod_{j=2}^{i-1} (1 + sr_j C_j)
\]

\[
= \sum_{i=2}^{n} \frac{1}{sC_i} \prod_{j=2}^{n} (1 + sr_j C_j)
\]
11.5 Calculate Settled Reference Voltage

\[ v_R H_{\text{den}} = H_{\text{num}} V_{\text{ref}} \]

\[
\begin{align*}
H_{\text{num}} &= sC_1 \prod_{j=2}^{n}(1 + sr_j C_j) \\
H_{\text{den}} &= (1 + sr_1 C_1) \sum_{i=2}^{n} sC_i \prod_{j=2, j \neq i}^{n}(1 + sr_j C_j) + sC_1 \prod_{j=2}^{n}(1 + sr_j C_j)
\end{align*}
\]

\[
\begin{align*}
H_{\text{num}} &= C_1 \prod_{j=2}^{n}(1 + sr_j C_j) \\
H_{\text{den}} &= (1 + sr_1 C_1) \sum_{i=2}^{n} C_i \prod_{j=2, j \neq i}^{n}(1 + sr_j C_j) + H_{\text{num}}
\end{align*}
\]

\[
A_1 \frac{d^{n-1}v_R}{dt^{n-1}} + A_2 \frac{d^{n-2}v_R}{dt^{n-2}} + \ldots + A_n v_R = B_1 \frac{d^{n-1}V_{\text{ref}}}{dt^{n-1}} + B_2 \frac{d^{n-2}V_{\text{ref}}}{dt^{n-2}} + \ldots + B_n V_{\text{ref}}
\]

\( n = \text{number of subranges} \)

parasitic \( C_p \Rightarrow \text{one extra branch} \}
\[
\Rightarrow \text{differential equation of order} \ n
\]

\( V_{\text{ref}} \text{ constant} \Rightarrow \frac{d^iV_{\text{ref}}}{dt^i} \]

\[
A_1 \frac{d^n v_R}{dt^n} + A_2 \frac{d^{n-1}v_R}{dt^{n-1}} + \ldots + A_{n+1} v_R = B_{n+1} V_{\text{ref}}
\]

Division by \( A_1 \Rightarrow \)

\[
\frac{d^n v_R}{dt^n} + a_2 \frac{d^{n-1}v_R}{dt^{n-1}} + \ldots + a_{n+1} v_R = b V_{\text{ref}}
\]
The reference voltage on the comparator input is

\[ v_R = K_0 V_{\text{ref}} + K_1 e^{r_{1t}} + K_2 e^{r_{2t}} + \ldots + K_n e^{r_{nt}}, \]

where \( K_i \) and \( r_{it} \) are found as demonstrated above.

**B.2 Calculate Settled Sampled Input Voltage**

The input signal is sampled to all sampling capacitors as shown in figure B.3.
11.5 Calculate Settled Sampled Input Voltage

![Equivalent circuit for sampling.](image)

**Figure B.3.** Equivalent circuit for sampling.

\[ v_1 = v_{in} \left( 1 - e^{-\frac{1}{r_S C_1 t}} \right), \]

The on-resistance of the switch is calculated as

\[ r_S = \frac{1}{\mu_n C_{ox} \frac{W}{L} (V_{dd} - v_{in} - V_t)} , \]

where \( \mu_n \) and \( C_{ox} \) are process parameters, \( W \) and \( L \) the transistor size, and \( V_t \) the threshold voltage. Superposition gives a total sampled voltage of

\[ v_{S,tot} = v_{in} \sum_{i} v_{in} \left( 1 - e^{-\frac{1}{r_S C_i t}} \right) \]

Since \( v_{in} \) is sampled to all \( n \) sampling capacitances, \( v_{S,tot} \) after settling is \( n \) times larger than \( v_{in} \). A division by \( n \) compensates for this. The sampled voltage is affected the same way as the reference voltage when applied to the RC network. According to the equation for \( v_R \), this means multiplication by \( K_0 \). The sampled voltage on the comparator thus becomes

\[ v_S = \frac{K_0 v_{in}}{n} \sum_{i=1}^{n} v_{in} \left( 1 - e^{-\frac{1}{r_S C_i t}} \right) \]

and the part used for comparisons in a certain subrange is
\[ v_S = \frac{C_i}{C_{tot}} \cdot K_0 v_{in} \sum_{i=1}^{n} v_{in} (1 - e^{-\frac{1}{\tau S C_i}}), \]

where \( C_i \) is the sampling capacitor of that subrange and \( C_{tot} \) the total sampling capacitance, \( \Sigma C_i \).

**B.3 Calculate Comparator Output Voltage**

The comparator output, see figure B.4, is given by

![Comparator](image)

Figure B.4. Comparator.

\[ v_{out} = \frac{1}{C_L} \int_{0}^{t} g_m (v_S - v_R) d\tau, \]

where \( t \) is the time available for settling. The sampled voltage, \( v_S \), is constant during comparisons for a certain output code.

\[ v_{out} = \frac{g_m}{C_L} \int_{0}^{t} \left[ v_S - \left( K_0 V_{ref} + K_1 e^{r_{i1} \tau} + K_2 e^{r_{i2} \tau} + \ldots + K_n e^{r_{in} \tau} \right) \right] d\tau = \]

\[ = \frac{g_m}{C_L} \left[ (v_S - K_0 V_{ref})t - \left( \int_{0}^{t} K_1 e^{r_{i1} \tau} d\tau + \int_{0}^{t} K_2 e^{r_{i2} \tau} d\tau + \ldots + \int_{0}^{t} K_n e^{r_{in} \tau} d\tau \right) \right] = \]

\[ = \frac{g_m}{C_L} \left[ (v_S - K_0 V_{ref})t - \frac{K_1}{r_{i1}} (e^{r_{i1} t} - 1) - \frac{K_2}{r_{i2}} (e^{r_{i2} t} - 1) - \ldots - \frac{K_n}{r_{in}} (e^{r_{in} t} - 1) \right] \]

\[ v_{out} = \frac{g_m}{C_L} \left[ (v_S - K_0 V_{ref})t - \frac{K_1}{r_{i1}} (e^{r_{i1} t} - 1) - \frac{K_2}{r_{i2}} (e^{r_{i2} t} - 1) - \ldots - \frac{K_n}{r_{in}} (e^{r_{in} t} - 1) \right] \]