An analysis of a digitally self calibrated parallel pipelined analog-to-digital converter

Venkata K. Navin
Iowa State University

Follow this and additional works at: https://lib.dr.iastate.edu/rtd

Part of the Digital Circuits Commons, Digital Communications and Networking Commons, and the Signal Processing Commons

Recommended Citation
https://lib.dr.iastate.edu/rtd/16972

This Thesis is brought to you for free and open access by the Iowa State University Capstones, Theses and Dissertations at Iowa State University Digital Repository. It has been accepted for inclusion in Retrospective Theses and Dissertations by an authorized administrator of Iowa State University Digital Repository. For more information, please contact digirep@iastate.edu.
An analysis of a digitally self calibrated parallel pipelined analog-to-digital converter

by

Venkata K. Navin

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Department: Electrical and Computer Engineering
Major: Computer Engineering
Major Professor: Marwan M. Hassoun

Iowa State University
Ames, Iowa
1996
Copyright © Venkata K. Navin, 1996. All rights reserved.
Graduate College
Iowa State University

This is to certify that the Master's thesis of
Venkata K. Navin
has met the thesis requirements of Iowa State University

Signatures have been redacted for privacy
# TABLE OF CONTENTS

ACKNOWLEDGMENTS ................................................................. xi

ABSTRACT .................................................................................. xii

1 INTRODUCTION ................................................................. 1

1.1 General Introduction ........................................................ 1

1.1.1 Organization ................................................................. 1

1.1.2 Overview ................................................................. 2

1.2 Definitions ........................................................................... 3

1.2.1 Nonlinearity Specifications ............................................. 3

1.2.2 Global Offset ............................................................... 3

1.2.3 Global Gain ................................................................. 4

1.2.4 Signal to Noise Ratio (S/N) ............................................ 4

1.2.5 Spurious Free Dynamic Range ....................................... 5

1.3 A/D Converter Architectures ............................................. 5

1.3.1 Flash Converters .......................................................... 5

1.3.2 Successive Approximation Architectures ...................... 5

1.3.3 Multi-step Converters ................................................... 7

1.3.4 Recycling Architectures ............................................... 8

1.3.5 Oversampling Architectures ......................................... 9

1.3.6 Pipeline Architectures .................................................. 10

1.3.7 Other Architectures ..................................................... 10

1.4 Conclusions ......................................................................... 11

2 DIGITAL CALIBRATION ....................................................... 12

2.1 Pipeline Architectures ...................................................... 12

2.2 Origin of Errors .................................................................. 13

2.3 Effect of these Errors on the Transfer Characteristic .......... 14
5 DESIGN OF COMPONENTS FOR PIPELINED CONVERTERS .......................... 72
  5.1 Sample/Hold .................................................. 72
  5.2 Design of the Operational Amplifier Used in the Sample/Hold ............. 73
    5.2.1 Two Stage Folded Cascode Architecture ................................ 75
    5.2.2 Single Stage Folded Cascode Design ..................................... 76
    5.2.3 Differential Sample/Hold Design ....................................... 77
    5.2.4 Architecture based on “Gain-Boost” Principle .......................... 78
  5.3 Design of Comparator Used in Flash ADC .................................... 79
  5.4 Conclusions .................................................... 82

6 SUMMARY, CONCLUSIONS AND FUTURE WORK ............................................ 83
  6.1 Scope ............................................................ 83
  6.2 Summary .......................................................... 83
  6.3 Conclusions ....................................................... 84
  6.4 Contributions ..................................................... 85
  6.5 Suggested Future Work .............................................. 85

APPENDIX THE MANUAL FOR SIMPAD ......................................................... 86

BIBLIOGRAPHY ............................................................. 93
LIST OF TABLES

Table 1.1 Various A/D architectures .............................................. 10
Table 2.1 Comparison of the two architectures ................................. 35
Table 4.1 Summary of all the results .............................................. 69
Table 5.1 Opamp specifications achieved ....................................... 77
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1.1</td>
<td>INL and DNL errors</td>
<td>4</td>
</tr>
<tr>
<td>Figure 1.2</td>
<td>A flash converter</td>
<td>6</td>
</tr>
<tr>
<td>Figure 1.3</td>
<td>Successive approximation architecture</td>
<td>7</td>
</tr>
<tr>
<td>Figure 1.4</td>
<td>DAC output waveform</td>
<td>7</td>
</tr>
<tr>
<td>Figure 1.5</td>
<td>Multi-step A/D converter</td>
<td>8</td>
</tr>
<tr>
<td>Figure 1.6</td>
<td>Recycling A/D converter</td>
<td>9</td>
</tr>
<tr>
<td>Figure 1.7</td>
<td>Sigma-Delta A/D converter</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.1</td>
<td>Pipelined A/D converter</td>
<td>13</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>A pipeline cell</td>
<td>14</td>
</tr>
<tr>
<td>Figure 2.3</td>
<td>Effect of errors on the transfer curve. Missing decision levels and codes are present</td>
<td>15</td>
</tr>
<tr>
<td>Figure 2.4</td>
<td>Multiply by 2 circuit</td>
<td>16</td>
</tr>
<tr>
<td>Figure 2.5</td>
<td>Code error correction: (a) uncalibrated transfer curve (b) digital code errors (c) transfer curve after code error calibration</td>
<td>18</td>
</tr>
<tr>
<td>Figure 2.6</td>
<td>Single converter stage</td>
<td>19</td>
</tr>
<tr>
<td>Figure 2.7</td>
<td>Pipelined stages</td>
<td>20</td>
</tr>
<tr>
<td>Figure 2.8</td>
<td>Single stage modified for calibration</td>
<td>22</td>
</tr>
<tr>
<td>Figure 2.9</td>
<td>Uncalibrated ADC transfer characteristic</td>
<td>27</td>
</tr>
<tr>
<td>Figure 2.10</td>
<td>ADC transfer characteristic after calibration by accuracy bootstrapping</td>
<td>27</td>
</tr>
<tr>
<td>Figure 2.11</td>
<td>INL for the calibrated ADC</td>
<td>28</td>
</tr>
<tr>
<td>Figure 2.12</td>
<td>DNL for the calibrated ADC</td>
<td>28</td>
</tr>
<tr>
<td>Figure 2.13</td>
<td>Effect of multiple calibrations on the transfer characteristic when $k$ is less than 1 Numbers indicate the calibration step</td>
<td>29</td>
</tr>
<tr>
<td>Figure 2.14</td>
<td>Calibration procedure to eliminate noise effects</td>
<td>30</td>
</tr>
<tr>
<td>Figure 2.15</td>
<td>Basic cell for the Karanicholas and Lee architecture</td>
<td>31</td>
</tr>
<tr>
<td>Figure 2.16</td>
<td>Pipeline ADC with digital calibration applied to the eleventh stage [2]</td>
<td>32</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>2.17</td>
<td>Digital calibration of higher stages [2]</td>
<td>32</td>
</tr>
<tr>
<td>2.18</td>
<td>Comparison of the two calibration algorithms a) Karanicholas and Lee's b) Sönnen and Geiger. Note that the analysis is the same for ΔV_{dac2}.</td>
<td>36</td>
</tr>
<tr>
<td>3.1</td>
<td>Model of the single stage</td>
<td>39</td>
</tr>
<tr>
<td>3.2</td>
<td>Single bow non linearity model</td>
<td>40</td>
</tr>
<tr>
<td>3.3</td>
<td>Double bow non linearity model</td>
<td>40</td>
</tr>
<tr>
<td>3.4</td>
<td>Flow chart for SIMPAD</td>
<td>42</td>
</tr>
<tr>
<td>4.1</td>
<td>Time interleaved multiple pipeline architecture</td>
<td>44</td>
</tr>
<tr>
<td>4.2</td>
<td>Two rank architecture</td>
<td>45</td>
</tr>
<tr>
<td>4.3</td>
<td>Mismatch effects on the frequency spectrum</td>
<td>46</td>
</tr>
<tr>
<td>4.4</td>
<td>Averaged parallel sampling for two paths</td>
<td>48</td>
</tr>
<tr>
<td>4.5</td>
<td>Quadrature mirror filtering</td>
<td>48</td>
</tr>
<tr>
<td>4.6</td>
<td>Uncalibrated multiple path 8 bit ADC output</td>
<td>50</td>
</tr>
<tr>
<td>4.7</td>
<td>Normalized multiple path 8 bit ADC output</td>
<td>51</td>
</tr>
<tr>
<td>4.8</td>
<td>INL plot for 8 bit ADC</td>
<td>51</td>
</tr>
<tr>
<td>4.9</td>
<td>DNL plot for 8 bit ADC</td>
<td>52</td>
</tr>
<tr>
<td>4.10</td>
<td>Uncalibrated multiple path 16 bit ADC output</td>
<td>52</td>
</tr>
<tr>
<td>4.11</td>
<td>Calibrated multiple path 16 bit ADC output</td>
<td>53</td>
</tr>
<tr>
<td>4.12</td>
<td>INL error after calibration for the 16 bit ADC</td>
<td>53</td>
</tr>
<tr>
<td>4.13</td>
<td>DNL error after calibration for the 16 bit ADC</td>
<td>54</td>
</tr>
<tr>
<td>4.14</td>
<td>FFT of the input</td>
<td>56</td>
</tr>
<tr>
<td>4.15</td>
<td>FFT of the output with no calibration</td>
<td>56</td>
</tr>
<tr>
<td>4.16</td>
<td>FFT of the output with the channels calibrated individually by accuracy bootstrapping</td>
<td>57</td>
</tr>
<tr>
<td>4.17</td>
<td>FFT of the output with global matching or normalization</td>
<td>57</td>
</tr>
<tr>
<td>4.18</td>
<td>FFT of the input with no opamp non linearity</td>
<td>58</td>
</tr>
<tr>
<td>4.19</td>
<td>FFT of the output with no calibration (no opamp non linearity)</td>
<td>59</td>
</tr>
<tr>
<td>4.20</td>
<td>FFT of the output with accuracy bootstrapped channels (no opamp non linearity)</td>
<td>59</td>
</tr>
<tr>
<td>4.21</td>
<td>FFT of the output with global matching or normalization (no opamp non linearity)</td>
<td>60</td>
</tr>
<tr>
<td>4.22</td>
<td>FFT of the input (opamp non linearity = 0.1%)</td>
<td>61</td>
</tr>
<tr>
<td>4.23</td>
<td>FFT of the output with no calibration (opamp non linearity = 0.1%)</td>
<td>61</td>
</tr>
</tbody>
</table>
Figure 4.24 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.1%) ........................................ 62
Figure 4.25 FFT of the output with global matching or normalization (opamp non linearity = 0.1%) ........................................ 62
Figure 4.26 FFT of the input (opamp non linearity = 0.01%) ........................................ 63
Figure 4.27 FFT of the output with no calibration (opamp non linearity = 0.01%) ........................................ 63
Figure 4.28 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.01%) ........................................ 64
Figure 4.29 FFT of the output with global matching or normalization (opamp non linearity = 0.01%) ........................................ 64
Figure 4.30 FFT of the input (opamp non linearity = 0.001%) ........................................ 65
Figure 4.31 FFT of the output with no calibration (opamp non linearity = 0.001%) ........................................ 65
Figure 4.32 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.001%) ........................................ 66
Figure 4.33 FFT of the output with global matching or normalization (opamp non linearity = 0.001%) ........................................ 66
Figure 4.34 FFT of the input (opamp non linearity = 0.001% and 3% component errors) ........................................ 67
Figure 4.35 FFT of the output with no calibration (opamp non linearity = 0.001% and 3% component errors) ........................................ 67
Figure 4.36 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.001% and 3% component errors) ........................................ 68
Figure 4.37 FFT of the output with global matching or normalization (opamp non linearity = 0.001% and 3% component errors) ........................................ 68
Figure 4.38 FFT of the input (opamp non linearity = 0.001% and 5% component errors) ........................................ 70
Figure 4.39 FFT of the output with no calibration (opamp non linearity = 0.001% and 5% component errors) ........................................ 70
Figure 4.40 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.001% and 5% component errors) ........................................ 71
Figure 4.41 FFT of the output with global matching or normalization (opamp non linearity = 0.001% and 5% component errors) ........................................ 71
Figure 5.1 Sample/hold cum multiplying DAC ........................................ 72
Figure 5.2 Differential version of sample/hold amplifier ........................................ 74
Figure 5.3  Internally compensated folded cascode CMOS opamp  ..........  75
Figure 5.4  Single stage folded cascode  ......................................  76
Figure 5.5  Output of opamp .......................................................  78
Figure 5.6  Cascoded gain stage with gain boosting .......................  79
Figure 5.7  Frequency response ...................................................  80
Figure 5.8  Schematic of the comparator .......................................  81
Figure 5.9  Time relation between φ1 and φ2  .................................  81
Figure 5.10  Final layout of the single cell and other typical components  82
Figure A.1  Flow chart for SIMPAD ..............................................  88
ACKNOWLEDGMENTS

I wish to express my sincere appreciation to my research advisor, Dr. Marwan Hassoun. I am grateful for his guidance and support throughout my masters program at Iowa State University. He introduced me to the field of A/D converters and was always there to help me. He provided an excellent atmosphere and facilities to perform my research. I also wish to thank Dr. William Black for being patient with all my “doubts” no matter what time of the day during the course of the last few months. He taught me a great deal about analog circuit design. I would also like to express my thanks to Wayne Dietrich and all the people at Texas Instruments for many technical discussions and for sponsoring a major part of this work. I would like to thank Dr. Richard Hester, Dr. Randall Geiger and Dr. Edward Lee for many discussions during the course of this work. I would also like to thank Hon Chi Ng and Tapas Ray for adding some of the FFT analysis and plotting capabilities in SIMPAD. Finally, I would like to thank professors like Dr. Sachin Sapatnekar at Iowa state University who taught me some great courses.

Throughout my masters program, I have had terrific colleagues in Lakshmikant Bhupathi (LKB), Jatan Shah, Jeff Echtenkamp, Naresh Maheshwari, Prasoon Surti, Saqib Malik, Tapas Ray and Jeff Shi. They were always helpful in doing the little things that were so important. I would also like to thank Subbarao, all the people in 203 campus avenue and all the guys who used to play basketball on Friday’s at the rec for making these two years very enjoyable.

Lastly, I would like to thank the people who made it all possible. My parents for always encouraging me and for making my ambitions their dreams. My grandparents for their support throughout and my aunt and her family for being supportive through my initial months at Iowa State University.
ABSTRACT

As present day signal processing systems increasingly use high performance digital techniques, there is a growing need for high speed/high resolution analog-to-digital converters (ADC's). New applications keep pushing for increasing conversion rates and simultaneously higher resolution architectures. Monolithic integrated circuit solutions in CMOS technology at such high speed/high resolution do not exist. The motivation for CMOS is the highest levels of integration possible - the ultimate goal is to integrate the A/D converter on the same IC as a digital signal processor in order to implement a complete analog/digital interface. The present trend in the design of these high speed/high resolution converters is dominated by pipelined architectures. Hence, there is also a need for simulation environments to design, test and characterize these converters.

The particular focus of this work is a resolution of 16 bits with an intent to push the speed as high as possible. The work proposes to achieve high speed by using the concepts of time interleaving and pipelining. The high resolution is to be achieved by calibration in the digital domain. Spurious free dynamic range (SFDR) is another important specification for ADC's used in communication applications such as radar. Hence, an important goal was to achieve a SFDR of the order of 90 dB. A functional simulator (SIMPAD - SIMulation of Multiple Pipelined Analog-to-Digital converters) which can be used in the design and testing of parallel pipelined analog-to-digital converters has also been developed. A technique to perform calibration of these multiple pipelines is also researched.

This work provides several contributions to the field of pipelined ADC's. A mathematical insight is provided to digital self calibration algorithms. The effect of multiple calibrations on the transfer characteristic, and a procedure to perform multiple calibrations were investigated. A simulation environment has been implemented which can be used in the design and characterization of pipelined and time interleaved A/D converters. The software has a user friendly interface and incorporated plotting and testing capabilities including linear ramp and FFT tests (single tone and double tone). Finally, a method termed as global normalization to perform calibration of parallel pipelines to eliminate spurious harmonics in the frequency spectrum has been explored.
1 INTRODUCTION

1.1 General Introduction

This thesis covers the study and analysis of a 16-bit parallel pipelined Analog-to-Digital converter for communication applications. With the increased usage of high performance digital techniques in signal processing, there is a growing need for increasingly fast and high resolution A/D converters. In the field of digital audio, the required conversion accuracy is high, but the conversion speed is limited (16 bits, $2 \times 20$ kHz signal bandwidth). In the field of image processing, the required accuracy is less, but the data conversion speed is high (8-10 bits, 5-20 MHz bandwidth). New applications keep pushing for increasing conversion rates and simultaneously higher resolution architectures. Monolithic integrated circuit solutions in CMOS technology at these high speed/high resolution do not exist. The motivation for CMOS is the highest levels of integration possible - the ultimate goal is to integrate the A/D converter on the same IC as a digital signal processor in order to implement a complete analog/digital interface. The particular focus of this work is a resolution of 16 bits with an intent to push the speed as high as possible. The work proposes to achieve high speed by using the concept of time interleaving and pipelining. The high resolution is to be achieved by calibration in the digital domain. The applications targeted by this high speed/high resolution A/D converter are mainly radar applications for communication.

1.1.1 Organization

Chapter 1 of this thesis provides a general introduction into A/D conversion principles. It starts by an introduction to the concepts involved in data conversion and then describes various terms used throughout the thesis. It ends by giving a broad overview of the various A/D converter architectures with a detailed overview of pipelined architectures. Chapter 2 goes into the concept of digital calibration. It begins with an introduction to conventional error correction techniques and then attempts to provide an insight into the origin of errors in various architectures. The effect of these errors on the transfer characteristic is also described. Finally, architectures and algorithms for digital self calibration are
introduced and a mathematical insight into how these self calibration techniques work is provided.

Chapter 3 describes a functional simulator (SIMPAD - SImulation of Multiple Pipelined Analog-to-Digital converters) that has been implemented. Initially, the interface, models used and the flowchart for the software are described. Then, testing and characterization of the converters by means of ramp inputs and FFT's is delved into. Chapter 4 goes into the design of a 16 bit A/D converter. An introduction to the concept of time interleaving is provided and various effects in the frequency spectrum are described. Then, various calibration techniques used in the literature are described. Finally, a technique for global calibration of the multiple pipelines is presented and various results of this technique are also presented. Chapter 5 mainly is an introduction to design techniques used for the design of pipelined A/D converters. The chapter goes into detail on the issues involved in designing the operational amplifier used in the sample/hold cum multiplying DAC circuit. Design of the comparator used for the Flash A/D is also discussed. An attempt to build some typical components is described. Finally, chapter 6 summarizes the work and discusses the scope for future work in this topic.

1.1.2 Overview

Analog-to-Digital (A/D) conversion is an important function to interface between "real-world" analog signals and digital computers. An analog-to-digital converter (ADC) is a device that takes an analog input signal and provides a corresponding digital output code. In other words, it first samples a continuous-time signal to give a discrete-time signal. Then, it quantizes the amplitude to one of a number of equally spaced levels and assigns a code to that label (usually a binary digital word).

An ADC defines a mapping between some input signal values which are normalized to a certain range and an output code. To each possible value at the input must correspond one and exactly one output code. This function which maps the input and the output is called the transfer function. The most common mapping and the one used throughout this thesis is the linear mapping, in which different output codes are assigned to nominally equal sections of the input range.

A linear mapping assigns codes such that input range segments of increasing magnitude would correspond to output codes of increasing or decreasing binary value. This concept is called monotonicity. Also, the concept of quantization introduces an irreversible error [1] termed as the quantization error. In an amplitude quantized system, using n binary weighted bits the number of quantization levels equals $2^n - 1$. Thus, a quantization error of $\pm \frac{1}{2^n}$ is introduced. Other errors are introduced in an ADC because of component non-idealities. These result in the transfer characteristic having a global offset and a gain which deviates from unity (ideal). These are discussed in more detail later.
1.2 Definitions

1.2.1 Nonlinearity Specifications

There is a wide variation in the literature in the definition of INL (Integral Non Linearity) and DNL (Differential Non Linearity) errors. The definitions presented below are the ones used throughout this thesis.

1.2.1.1 Integral Nonlinearity (INL)

Integral Non Linearity is the maximum deviation of the actual transition points in an ADC’s transfer characteristic from the straight line drawn between the end points (first and last code transitions). Another definition for the INL uses the best fit line of all the conversion points as the transfer characteristic instead of the line between the two end points.

1.2.1.2 DNL (Differential Non Linearity)

DNL is defined as the difference between the ideal bin width of the converter and the actual width of each bin. A bin is defined as the set of input voltages yielding the same output code. Therefore,

\[ inl_i = \sum_{k=0}^{i} dnl_k \]

Note that \( i \) stands for the \( ith \) bin. Figure 1.1 graphically shows the INL and DNL errors as described above.

The INL and DNL errors are very significant characteristics of the ADC. They are both measures of linearity. The DNL reflects the resolution of the ADC whereas the INL reflects the accuracy of the ADC. Thus, these non-linearity measures can be used to determine the type of ADC used for different applications. For example, if we needed a high resolution ADC for a video application, we could use the DNL as a measure of its resolution.

1.2.2 Global Offset

Input amplifiers, output amplifiers, and comparators in practical circuits inherently have a built-in offset voltage and offset current. This offset is caused by the finite matching of components. The offset results in a non-zero input or output voltage, current or digital code although a zero signal is applied to the input of the converter.
1.2.3 Global Gain

As described above, various components in the converters have non-ideal gains. This results in the overall transfer characteristic of the converter having a gain other than unity. The gain of the transfer characteristic is defined as the slope of the line drawn between the the two endpoints. That is the points corresponding to the first and last transitions in Figure 1.1.

1.2.4 Signal to Noise Ratio (S/N)

Dynamic specifications are specifications used to determine the applicability of the ADC to certain signal processing applications like digital audio or video. S/N ratio is one of the most important dynamic specifications of the converter. It depends on the resolution of the converter and automatically includes specifications of linearity, distortion, sampling time uncertainty, glitches, noise and settling time. It ideally follows the theoretical formula [1]:

\[ S/N_{max} = 6.02n + 1.76dB \]

\( n \) stands for the number of bits. The S/N ratio is calculated for a sine wave input with a maximum amplitude. Another requirement is that the ratio between the frequency of the sine wave, and the sampling frequency should be irrational.
1.2.5 Spurious Free Dynamic Range

An important specification, especially for converters targeted to communication applications is the spurious free dynamic range (SFDR). This is a characteristic of the frequency domain. SFDR is defined as the ratio in decibels between the magnitude of the fundamental component and the magnitude of the largest harmonic or inter-modulation product. It is an indication of the dynamic operating range of the ADC. It is usually measured by a single tone or two tone test which are described later on.

1.3 A/D Converter Architectures

Since architectures to perform data conversion are numerous, a few important schemes to perform A/D conversion are described in this section.

1.3.1 Flash Converters

This is probably the most straightforward way to implement an A/D converter. Due to its inherent simplicity, this type of converter can be made extremely fast and impressive results have been reported in the literature [2] [3]. Figure 1.2 is a block diagram of an $m$-bit flash converter. The circuit consists of $2^m$ comparators, a resistor ladder comprising $2^m$ equal segments, and a decoder. The ladder subdivides the main reference into $2^m$ equally spaced voltages, and the comparators compare the input signal with these voltages. For example, if the analog input is between $V_j$ and $V_{j+1}$, comparators $A_1$ through $A_j$ produce ones at their output while the rest generate zeroes. Consequently, the comparator outputs constitute a thermometer code [4], which is converted to binary by the decoder. The main disadvantages of this architecture are that it requires a large chip area and it consumes a lot of power. In fact, the chip area increases exponentially with bit resolution (because the number of comparators and resistors increase exponentially).

1.3.2 Successive Approximation Architectures

Successive approximation employs a “binary search” algorithm in a feedback loop including a 1-bit A/D converter. They are used to achieve relatively high resolution (13-15 bits) at limited conversion rates (100 kHz - 1MHz) [5] [6]. Figure 1.3 illustrates this architecture, which consists of a front-end SHA (sample/hold amplifier), a comparator, a pointer (shift register), decision logic, a decision register, and a DAC (Digital to Analog Converter). The pointer points to the last bit changed in the decision
register, and the data stored in this register is the result of all the comparisons performed in the present conversion period.

During a binary search, the circuit halves the difference between the held signal, $V_H$, and the DAC output, $V_D$, in each clock cycle. The conversion proceeds as follows. First, both the pointer and the decision register are set to midscale (100...0) so that the DAC produces the midscale analog output. The comparator is then strobed to determine the polarity of $V_H - V_D$. The pointer and decision logic direct the logical output of the comparator to the most significant bit (MSB) in the decision register. Thus, if $V_H > V_D$, the MSB of this register is maintained at 1, and if $V_H < V_D$, it is set to 0. Subsequently, the pointer is set to 110...0 and the next bit in the decision register is set to 1. After the DAC output has settled to its new value, the comparator is strobed again and the above sequence is repeated. Figure 1.4 illustrates the DAC output waveform in a typical conversion period. Initially, the DAC output $V_D$ is at $\frac{V_{FS}}{2}$ where $V_{FS}$ stands for the full scale voltage. It then reduces to $\frac{V_{FS}}{4}$ and another value both of which are bigger than $V_H$. It then becomes smaller than $V_H$ and then increases until it is accurate to $n$ bits. For a resolution of $M$ bits, the successive approximation architecture is at least $M$ times slower than its flash counterpart. But it offers several advantages. The comparator offset only affects the offset and does not affect the linearity. Second, it does not require an explicit subtracter and lastly, the circuit complexity and power dissipation are less than other architectures.
1.3.3 Multi-step Converters

The main disadvantage of the elementary flash converter is the large component count (and hence chip area and power consumption) required to realize higher resolution schemes. To circumvent this problem, multi-step converters have been implemented [7] [8].

A multi-step converter divides the useful input signal range into a number of sub-ranges using a comparator bank, like in the flash converter. The difference is that the number of sub-ranges is less than the total number of desired bins ($2^n$) where a bin is defined as the set of input voltages yielding the same output code. Next, the comparator output codes are used to address a bank of voltages (or currents, depending on the converter type) to be subtracted from the input signal. This operation (DAC operation) calculates an analog “residue” or remainder, which reflects the portion of the input signal that exceeds the lower boundary of the signal section in question. Naturally, the residue of the
conversion will have a range that is smaller than the range of the input signal.

The second step of the conversion consists in taking the residue from the first step, and converting it in a way similar to the first step. If the circuitry implementing the second step (the second "stage") does not have a gain stage at its input, the range of the signal to be converted will be significantly smaller than the range of the first stage. As a result, the comparator reference levels must be scaled accordingly. This configuration is often referred to as a "sub-ranging" converter. However, if the second stage has an input amplifier that amplifies the signal to the same nominal range as the input signal of the previous stage, the same comparator reference levels could be used. This can be implemented as a number of nominally identical stages, cascaded together. Figure 1.5 depicts a typical 2-step A/D converter. Two step architectures are a tradeoff between speed on the one hand and power and area on the other.

### 1.3.4 Recycling Architectures

A recycling A/D is closely related to a pipelined scheme. It is another implementation of a multi-step approach and it has been used to obtain relatively high resolutions on a relatively limited silicon area [9]. A recycling converter implements the multi-step algorithm by taking an input signal, generating a local code and calculating a residue. That residue is normally fed to another stage, which can be nominally identical to the first one if the correct gain is implemented at the input.

A recycling converter realizes the same idea, using only one stage. The output of that stage (the residue) is fed back to the input of the same stage, and the process is repeated as many times as needed to obtain the desired number of bits. Then, the input is connected to the external signal again, and the
process can start over for the next sample. Note that the timing is different from the multi-step case. Figure 1.6 illustrates a typical recycling converter. The advantage of this architecture is that there is less area and power consumption. On the other hand, the timing is more complex and the speed is $N/M$ times slower than a non-recycling architecture. Note that $N$ is the total number of bits and $M$ the number of bits in one stage.

1.3.5 Oversampling Architectures

Oversampling is another way to trade-off conversion speed against resolution. One could use a very fast, low resolution converter to take a large number of samples of an analog signal. The corresponding digital outputs are acquired at the desired actual speed. The process is called oversampling since more samples are taken of the analog signal than needed to achieve the eventual desired output (digital) data rate.

The digital values can be combined into successive groups, and every time, some kind of average of the values within each group can be computed, using a digital filter. The averaging operation can increase the accuracy of the conversion, provided the input signal varies very slowly compared to the actual sampling rate. An additional advantage of this method is that some of the noise present in the analog signal can be reduced as well. The most common class of oversampling converters are the sigma-delta converters [10]. A basic scheme of these converters is shown in Figure 1.7. The main advantage of this architecture is the simplicity of the analog components though it is limited to high accuracy, low frequency applications. Stability is also a concern in higher order sigma-delta converters.
Pipeline architectures are the popular in the present day architectures. High speeds which are comparable to the flash architectures are attained by them. Since the architecture targeted in this work is a pipelined one, it is explained in more detail in the next chapter.

Other Architectures

In addition to these architectures, there are a number of architectures to perform data conversion. These include folding and interpolating A/D converters, single-slope, dual-slope type A/D converters, algorithmic A/D converters, interleaved converters and more. A description of the various kinds of A/D architectures can be found in [11]. Table 1.1 summarizes the advantages and disadvantages of the architectures discussed.
1.4 Conclusions

This chapter consisted of a brief review of A/D converters in general. The next chapter will delve more deeply into pipeline architectures and the errors which occur in those architectures. It will look into the topic of digital calibration and investigate architectures and algorithms to perform digital self calibration.
2 DIGITAL CALIBRATION

One of the inherent problems affecting the design of high-performance A/D converters is the fact that a highly accurate device is to be made from much less accurate components like capacitors, voltage sources etc. For example, the accuracy of matching capacitors is around one in thousand which corresponds to just 10 bit accuracy whereas the application might demand an ADC of 16 bit accuracy. The integral linearity of data converters usually depends on the matching of integrated resistors, capacitors, or current sources. For high resolution, means must be sought to correct these linearity errors. This is often accomplished by effective matching of individual errors or by correcting the overall transfer characteristic.

2.1 Pipeline Architectures

Pipeline A/D converters present advantages compared to flash or successive approximation techniques because potentially high resolution and high speed can be achieved at the same time. The functional block diagram for a pipelined multistage A/D converter is shown in Figure 2.1. The pipelined approach has been used extensively for medium to high speed in CMOS and BiCMOS technologies [12]-[20].

In a pipelined converter, each stage is preceded by a sample/hold amplifier which samples the input signal and then holds it constant for a certain time while the conversion is being performed. After the comparator outputs settle, a voltage determined by the comparator outputs is subtracted from the signal and the residue is calculated (DAC operation). This residue is then sampled by the sample/hold amplifier of the next stage, and held for the time of its conversion. Therefore, the first stage operates on the most recent input sample while the second stage operates on the gained-up residue from the previous sample etc. The sample/hold therefore functions as an analog pipeline latch.

The sequence of operations in each stage is synchronized by a multi-phase clock. The clocking scheme is determined in such a way as to realize a pipeline, in the same sense as pipelined logic is organized in digital circuits. The input signal can be applied to the first stage in line, as soon as the
second stage has sampled the output of the first one. As a result, it is not necessary to wait for a signal to travel through the whole pipeline before applying the next signal and thus, the overall throughput is increased considerably. All stages in a pipelined converter are nominally identical, and all of them have an input sample/hold amplifier in order to make the pipelined operation possible. However, the sample/hold of the first stage must be able to sample a rapidly changing external input signal without noticeable distortion, while the sample/hold amplifiers of the other stages only see the DC output level (residue) of the previous stage. As a result, the design of the first stage usually requires extra precision.

In the implementation of pipelined multi-stage A/D converters, digital error correction/over-range detection is usually employed with 1 bit of redundancy in each stage. This relaxes the offset requirements of the comparators. This is the primary emphasis of this chapter.

### 2.2 Origin of Errors

Since, the architecture targeted is a pipelined architecture, the following sections deal with errors in the components of the pipeline cell. As has been described, the main components of a pipeline cell are a flash A/D converter, a D/A converter, a summer and a sample/hold amplifier. Figure 2.2 shows a typical pipeline cell. The errors which occur in this pipelined cell can be classified as Flash ADC errors,
Flash ADC errors are due to non-idealities in the components used to make the Flash A/D converter. These errors occur mainly due to comparator offset and reference level errors. Reference level errors are usually due to errors in the resistor chain. Typically, in CMOS technology, matching of resistors and capacitors is accurate to 10 bits resulting in large mismatches when the required resolution is of the order of 16 bits.

DAC errors are the errors that occur in the reconstructed input from the digital-to-analog converter (DAC). The main sources of these errors are reference levels and switch feed-through effects. Since the DAC is usually a set of references selected by clocked switches, errors occur due to clock feed-through in the switches and due to errors in the reference levels. In fact, as the process dimensions keep shrinking, the effect of feed-through increases considerably as the parasitic capacitances are more dominant.

Gain errors mainly occur because of capacitor mismatch. The sample/hold amplifier is usually implemented as a switched capacitor gain stage with the gain being the ratio of two capacitors. Since, there is almost always some capacitor mismatch, this contributes to the gain error. Note that offsets in the interstage amplifier (sample/hold) also introduce non-linearity errors, but the effect is comparable to that of a DAC error and should not be treated separately. Finally, thermal noise sampled onto the sampling capacitors, commonly known as $kT/C$ noise, also introduces errors at resolutions higher than 10 bits.

2.3 Effect of these Errors on the Transfer Characteristic

To illustrate the effect of these errors on the transfer characteristic, consider the case of a 1 bit per stage converter [22]. Figure 2.3 shows the ideal residue plot and the effects of principal errors on the
residue plot. The dashed line represents the reference boundary that passes through coordinates $\pm V_{ref}$ along the $V_{out}$ axis and $\pm V_{ref}$ along the $V_{in}$ axis. Charge injection offset (clock feed-through) causes a vertical shift of the residue plot. Near the major carry transition point (the point where the carry is propagated all through for example 011...1 to 100...0), the residue exceeds the reference boundary, resulting in missing decision levels. Near the major carry transition point, the residue minimum does not extend to $-V_{ref}$, resulting in a gap from the minimum to the reference boundary; missing codes result. This is because the full input range of the remaining pipeline section is not accessed. Comparator offset causes a shift of the major carry transition point. This leads to the residue exceeding the reference boundary as well as to a gap in the reference boundary. Again, missing decision levels and missing codes, respectively result. Finally, capacitor mismatch as indicated causes the residue to exceed the reference boundary near the major carry transition point, resulting in missing decision levels. Capacitor mismatch could also lead to a gap from the residue extrema to the reference boundary near the major carry transition point, resulting in missing codes.
2.4 Calibration in the Analog Domain

Architectures have been proposed which try to get around the accuracy problem by tuning the critical components to their ideal values with the help of analog techniques. Since, high resolution, high speed A/D converters typically employ multi-step pipelined architectures, two prominent error correction techniques targeting these architectures are described below.

2.4.1 Capacitor Error Averaging

Since the sample/hold amplifier in the pipeline cell does a multiply by a power of 2 function, the accuracy depends on capacitor matching and opamp gain. In order to suppress the effect of capacitor mismatch, capacitor error averaging can be used [16]. Figure 2.4 depicts the multiply by 2 circuit in 2 modes. For an ideal opamp, the output is

\[ V_{out1} = V_{in}(1 + \frac{C_2}{C_1}) \]

Now, if the function was repeated with \( C_1 \) and \( C_2 \) interchanged,

\[ V_{out2} = V_{in}(1 + \frac{C_1}{C_2}) \]

In the capacitor averaging technique, both these outputs are sampled and averaged. If there is a mismatch of \( \Delta C \) between the two capacitors, the resulting output is then

\[ V_{avg} = V_{in}(2 + \frac{\Delta C^2}{2C^2}) \]

Since the error is now proportional to the square of the mismatch and not linearly dependent, the effect of capacitor mismatch is considerably reduced.
2.4.2 Capacitor Trimming

In this technique, the capacitors are typically implemented as a capacitor array and then individual capacitors are trimmed by a laser in order to achieve acceptable matching [23]. In practice, the individual capacitors must be small and must offer sufficient trimming range. This technique is also very costly to implement.

2.5 Digital Calibration

Digital calibration techniques quantize nonlinearity errors (deviations of the transfer characteristic from the ideal curve) in digital forms and remove these pre-measured errors during normal conversion [23] [24]. They reduce the need for precision analog components and leave the analog portion of the ADC intact by moving all calibration hardware into the digital domain except for switches used during calibration. Digital error calibration techniques can be categorized into two groups based on their error measuring methods. In the discussion below, output code stands for the digital representation of the analog input as determined by the A/D converter. In one group of techniques, individual bit nonlinearities usually appearing as capacitor mismatch errors are digitally measured and the total error for a given output code, the code error, is computed from individual bit errors during normal conversion [19]. The other group of techniques store digital code errors directly and eliminate the digital code-error computations during normal operation. It is described in the next section. The former calibration technique requires less digital memory but more digital computations during normal conversion while the latter technique requires more digital memory but less digital computations. Both these techniques are very suitable for multi-step converters. The later method, called code error calibration, when properly implemented, simultaneously reduces most nonlinearity errors including DAC nonlinearities. The only disadvantage is an exponential increase in the number of error measurements as the number of calibrated bits increases (since an n-bit converter requires $2^n$ digital memory words for storing the errors). Techniques have been proposed which optimize intermediate solutions trading the number of error measurements with the number of digital computations [20].

2.5.1 Code Error Correction

The ADC nonlinearity is bound by mismatch of components at major code transition points [19]. A segment is the set of continuous points which are linear. Usually segments occur between major code transition points. This situation is illustrated conceptually in Figure 2.5(a) where each segment is
assumed to be linear. If less significant digital output codes are grouped as segments and each segment
is dislocated by a certain amount from the ideal straight line as in Figure 2.5(a), the digital amounts of
dislocation measured from the ideal line can be defined as code errors as shown in Figure 2.5(b). That
is, each dislocated segment can be moved back to the straight line by digitally subtracting the amount
of dislocation from each digital output occurring in that range as in Figure 2.5(c). The amounts of
dislocation are directly measured during the calibration cycle and stored in memory. These code errors
are addressed during the conversion and subtracted digitally from the uncalibrated digital outputs to
give the resulting calibrated output.

2.5.2 Redundancy

In a practical pipelined ADC, enough inherent redundancy can always be built into the different
stages to make the global structure insensitive to a certain amount of error on the local flash ADC
levels (for each pipeline cell). The process consists of extending the input range of the next stage with
respect to the output range of the current stage. As a result, the residue of one stage can never over
drive the input of the next stage and a certain amount of nonlinearity is prevented.
2.6 Architectures and Algorithms for Digital Self Calibration

Self calibration is a powerful concept which utilizes the components present within the ADC to measure its own errors and correct them. Since, the goal is to achieve an ADC with the calibration circuitry on chip, two promising digital self calibration algorithms [25] [22] to achieve the required accuracy were investigated.

2.6.1 Accuracy Bootstrapping

2.6.1.1 Mathematical Description

The schematic of a one converter stage is shown in Figure 2.6. Let the total number of stages be L and let \( l \) denote the \( l \)th converter (see Figure 2.7). The incoming signal \( V_{in} \) is compared against a number of reference or ADC levels, using a flash converter. The comparator outputs provide a rough digital representation of the input voltage, in thermometer format. Let \( b \) denote the number of bits/stage of the pipeline. \( b \) is a number such that \( 2^b \geq M \). If we call the digital output code \( c_l \), and there are M comparators, the \( M + 1 \) possible outputs are

\[
C_l = 0 \quad \text{for} \quad V_{in} < V_{ref}[0]
\]
Figure 2.7 Pipelined stages

\[
V_i^{\text{ref}}[0] \leq V_i^{\text{in}} \leq V_i^{\text{ref}}[1]
\]

\[
\vdots
\]

\[
M \quad \text{for} \quad V_i^{\text{ref}}[M - 1] \leq V_i^{\text{in}}
\]

where

\[
V_i^{\text{ref}}[i] = V_i^{\text{ref}}[0] + \frac{iV_{\text{ref}}}{2^{b-1}}
\]

Let \(V_i^{\text{res}}\) represent the output of a stage i.e the residue and \(V_i^{DAC}[c_i]\) the output from the DAC (reconstructed input). For any given \(c_i\), a general equation can be written for the residue \(V_i^{\text{res}}\) as a function of \(V_i^{\text{in}}\).

\[
V_i^{\text{res}} = (V_i^{\text{in}} - V_i^{DAC}[c_i])A_i
\]

Rearranging,

\[
V_i^{\text{in}} = V_i^{DAC}[c_i] + \frac{V_i^{\text{res}}}{A_i}
\]

From the pipeline structure,

\[
V_{i-1}^{\text{in}} = V_i^{\text{res}}
\]

But due to the pipelining, the input voltage for that stage can be written as

\[
V_i^{\text{in}} = V_i^{DAC}[c_i] + \frac{V_{i-1}^{DAC}[c_i - 1]}{A_i} + \frac{V_{i-1}^{\text{res}}}{A_iA_{i-1}}
\]

Note that this equation is derived assuming that there are no hysteresis and noise effects and the stage parameters are time and signal invariant [25]. In the discussion for the later part we will assume that one redundant comparator per stage is used. This can be seen in Figure 2.8.

The previous equation can be used to write \(V^{\text{in}}\) as

\[
V^{\text{in}} = V_{L-1}^{DAC}A^L + V_{L-2}^{DAC} \frac{A^{L-1}}{A_{L-1}} + \ldots + V_0^{DAC} \frac{A^{L-1}}{A_{L-1}\ldots A_0} + \frac{V_0^{\text{res}}}{A_{L-1}\ldots A_0}
\]
which can be written in terms of the weights (see next section for definition of weights) as

\[
Digital \ output = [[[[W_{L-1}A + \frac{W_{L-2}}{A}A + \frac{W_{L-3}}{A^2}A + \ldots}A + \frac{W_0}{A^L}A
\]

Comparing the above two equations, we get

\[
W_i^c = \frac{V_i^{DAC}[c]A^{(L-1-i)}}{A_{L-1} \ldots A_{i+1}}
\]

2.6.2 Explanation of the Architecture of a Single Cell

The architecture that calculates the conversion result according to the above equation is given in Figure 2.8. Depending on the value of \(cal\) the input \(V_{in}\) or a fixed voltage \(V_{fix}\) can be input to the pipeline stage. The output of the flash is used to select some weights (\(w[0], w[1] \) and \(w[2]\)) which are stored in a RAM. These weights are used as weighted coefficients whose sum is used to determine the digital output code. As can be seen in the figure, the output of the RAM is divided by \(2^L\) and added to the output of the previous stage. This is then multiplied by 2 to give the output of the present stage. The digital hardware implements the equation for the input which was derived in the previous section. Note that if \(A = 2\), the multiplication and division can be done by a simple shifting operation. The advantage of this scheme is that the numeric range for each stage is identical and the digital hardware can thus be duplicated from stage to stage. Note that two voltage sources or increments which can be termed \((\Delta V_{DAC}[1])\) and \((\Delta V_{DAC}[2])\) both of which are shown as being equal to 0.5 volts are present in the figure. Note that

\[
V_i^{DAC}[1] = V_i^{DAC}[0] + \Delta V_i^{DAC}[1]
\]
\[
V_i^{DAC}[2] = V_i^{DAC}[0] + \Delta V_i^{DAC}[1] + \Delta V_i^{DAC}[2]
\]

In our case, \(V_i^{DAC}[0]\) is -0.25 V, \(\Delta V_i^{DAC}[1]\) and \(\Delta V_i^{DAC}[2]\) are 0.5 V ideally. These voltage sources or increments are switched on by the two switches shown in the figure. Note that 3 possible results/stage are possible. Thus the bits per stage are 2 but only 3 out of the possible 4 codes are used. Also, since it has 0.25 V over range capability (as shown in Figure 2.8), it is in effect a 1.5 bits/stage architecture with 0.5 bits overlap between stages. The procedure followed for calibration is described in the next section.

2.6.2.1 Procedure for Calibration

The basic idea of accuracy bootstrapping is to individually measure all the DAC levels of each converter stage, using the remaining stages of the pipeline [21]. The measurements are used to update...
Figure 2.8 Single stage modified for calibration

the look-up tables of that stage, and the process is repeated until each stage has been calibrated. The algorithm is very powerful because it uses the data path already present to perform the calibration. The following section describes the procedure for the calibration. Figure 2.8 shows the basic stage modified for calibration. Experimentally, it has been observed that the calibration results in an iterative numerical problem that converges to the result very fast. The procedure is as follows:

1. The weights are initialized to their nominal values i.e. the values assuming no error in the component values.

2. The last stage is calibrated first. The analog input of that stage is held at a fixed potential, while none of the voltage increments (voltage sources in Figure 2.8) are enabled. The output of this stage is connected to the input of the first stage and this circular structure (see Figure 2.7) is used to determine the conversion result or digital code (let's call it \( N_1 \)).

3. The first voltage increment for the last stage is enabled \( \Delta V_0^{DAC}[1] \) by switch S1. The same L-1 stages of the circular structure are used to determine the new conversion result \( N_2 \). Similarly \( N_3 \) is determined (after enabling the second voltage increment (switch S2) instead of the first).

4. The weights are then calculated as follows: \( W_0[0] \) is fixed to its nominal value. The rest of the weights are calculated as follows:

\[
W_0[l] = W_0[0] + \left( \frac{N_1 - N_2}{A} \right)
\]
All the original (nominal) \( W_0[c] \) values of stage 0 are now replaced by the newly determined ones. This concludes the calibration of stage 0.

(5) The procedure used on stage 0 is repeated to calibrate stage 1. The voltage increments of stage 1 are measured, using the converter formed by stages 0, L-1, L-2, ..., 2, and the comparator stage of stage 1. The new values of \( W_1[c] \) are calculated and updated. The same procedure is repeated to calibrate the stages \( l = 2, ..., L-1 \) i.e the voltage increments of stage \( l \) are measured, using the converter formed by stages \( l-1, l-2, ..., 0, L-1, ..., l \). The previously calibrated stage is always the first (most significant) stage in the converter used to calibrate the next one.

(6) After one iterative calibration cycle through all the stages, one could repeat the procedure to further refine the weights and reduce the noise effects (see subsection 2.6.2.7). We propose a different scheme to average out the noise effects and this is described later on in this section.

2.6.2.2 How the Algorithm works

In the discussion that follows \( V_i^{DAC}[0] \) stands for the zero DAC level. \( \Delta V_i^{DAC}[1] \) and \( \Delta V_i^{DAC}[2] \) stand for the incremental voltage sources that are used in the DAC level generator (see Figure 2.8). Note that 0.25 is the value of the fixed voltage that is used for calibration. The accuracy bootstrapping algorithm first takes the value of \( 0.25 - V_i^{DAC}[0] \) and converts this. Then the first DAC voltage source is enabled and subtracted and this result is converted. Therefore,

\[
N_1 = (0.25 - V_i^{DAC}[0])A_f
\]

\[
N_2 = (0.25 - V_i^{DAC}[0] - \Delta V_i^{DAC}[1])A_f
\]

and finally the last source is enabled which gives

\[
N_3 = (0.25 - V_i^{DAC}[0] - \Delta V_i^{DAC}[2])A_f
\]

In the first analysis, let us consider the ADC to be perfect to full accuracy. Then the weights would be calculated as follows: For stage 0 (since the calibration starts from the lsb stage),

\[
W_0[1] = W_0[0] + \left( \frac{N_1 - N_2}{A} \right)
\]

and

\[
W_0[2] = W_0[1] + \left( \frac{N_1 - N_2}{A} \right)
\]
Therefore if there was no error in calculating the final result in the ADC,

\[
\frac{N_1 - N_2}{A} = \frac{\Delta V_0^{DAC}[1]A_0}{A}
\]

and

\[
\frac{N_1 - N_3}{A} = \frac{\Delta V_0^{DAC}[2]A_0}{A}
\]

Hence, we are overestimating \(W[l] - W[0]\) by a factor of \(\frac{A_0}{A}\). But ideally (refer to section on the mathematical description),

\[
W_i'[c] = \frac{V_i^{DAC}[c]A(L-1-l)}{A_{L-1}...A_{l+1}}
\]

where \(W_i'[c]\) is the ideal weight of the \(l\)th stage. Therefore

\[
W_i'[1] - W_i'[0] = \frac{\Delta V_0^{DAC}[1]A(L-1-l)}{A_{L-1}...A_{l+1}} \text{ ideally}
\]

For stage 0,

\[
W_0[1] - W_0[0] = \frac{\Delta V_0^{DAC}[1]A(L-1)}{A_{L-1}...A_1}
\]

Therefore, we are overestimating \(W_0[1] - W_0[0]\) by a factor of \(\frac{A_0...A_{L-1}}{A^L} = \frac{A_0...A_{L-1}}{A^L}\) if the ADC were not ideal, which is the case. In fact, in the beginning the weights are the nominal values and therefore there is an error (represented by \(\Delta\)) introduced in the accuracy of the measurement of \(\Delta V_0^{DAC}[1]A_0\). For stage 0,

\[
N_1 - N_2 = W_{L-1}[2] - W_{L-1}[0] + \Delta_1
\]

This can be lumped as a percentage error \(\epsilon_1\) when calculating \(N_1 - N_2\) and \(\epsilon_2\) when calculating \(N_1 - N_3\). Thus, \(\Delta_1\) is due to some other lower order terms and therefore, \(N_1 - N_2\) can be expressed as \(\Delta V_0^{DAC}[1]A_0(1 + \epsilon_1)\). Therefore for stage 0, \(W_0[1] - W_0[0]\) is overestimated by \(\frac{A_0...A_{L-1}}{A^L}(1 + \epsilon_1)\)

Summarizing, we note that there are three terms contributing to this overestimation. These are

1. \(1 + \epsilon_1\) due to the non ideal or nominal weights, \(\frac{A_0}{A}\) due to the fact that we divide \(N_1 - N_2\) by \(A\) instead of \(A_0\) and \(\frac{A(L-1-l)}{A_{L-1}...A_{l+1}}\) which is due to the difference from the ideal \(W_0[1] - W_0[0]\).

In general for any stage \(l\), we can write these contributions as:

Firstly, \(1 + \epsilon_1\) due to the non ideal or nominal weights. The second term needs a little explanation. For stage 1, we have a factor \(\frac{A_0}{A}\) due to the fact that we divide \(N_1 - N_2\) by \(A\) instead of \(A_1\). But we have already overestimated \(W_0[2] - W_0[0]\) by \(\frac{A_0}{A}\) when we calibrated stage 0 and since this is the predominant term when we calculate \(W_1[1] - W_1[0]\) and \(W_1[2] - W_1[0]\), the overall overestimation is \(\frac{A_1A_0}{A^L}\) for stage 1. This can be generalized for any stage \(l\). And lastly, \(\frac{A(L-1-l)}{A_{L-1}...A_{l+1}}\) which is the difference from the ideal \(W_1[1] - W_1[0]\).
Thus, in conclusion, the overestimation of
\[ W_i[1] - W_i[0] = \frac{A_0 \ldots A_L - 1}{A_L}(1 + \varepsilon_i) \]
\[ W_i[2] - W_i[0] = \frac{A_0 \ldots A_L - 1}{A_L}(1 + \varepsilon_2) \]

Note that \( \varepsilon_1 \) and \( \varepsilon_2 \) are different because \( \Delta V_i^{DAC}[1] \) and \( \Delta V_i^{DAC}[2] \) are different. We will now use the derivations above to show that Accuracy bootstrapping linearizes the ADC transfer characteristic. We will also show that as more calibrations are performed, the curve either shifts upward or downward (i.e., slope and offset either increase or decrease) depending on whether the factor \( \frac{A_0 \ldots A_L - 1}{A_L} \) is smaller or greater than unity. \( \frac{A_0 \ldots A_L - 1}{A_L} \) is henceforth referred to as \( k \) or Gain Ratio.

### 2.6.2.3 How the Curve is Linear

Any result given by the ADC can be written as
\[
\sum_{\text{some terms}} W_i[0]\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} W_i[1]\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} W_i[2]\left(\frac{1}{A_i}\right)
\]
which can be simplified to
\[
\sum_{\text{all terms}} W_i[0]\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} (W_i[1] - W_i[0])\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} (W_i[2] - (W_i[0])\left(\frac{1}{A_i}\right)
\]

Now, from the previous discussion we know that \( W_i[1] - W_i[0] \) and \( W_i[2] - W_i[0] \) are each overestimated by \( \frac{A_0 \ldots A_L - 1}{A_L}(1 + \varepsilon_i) \). Hence, we can write
\[ W_i[1] - W_i[0] = k(1 + \varepsilon_1) \quad \text{and} \quad W_i[2] - W_i[0] = k(1 + \varepsilon_2) \] where \( k = \frac{A_0 \ldots A_L - 1}{A_L} \)

Therefore any result can be expressed as
\[
\sum_{\text{all terms}} W_i[0]\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} k(W_i[1] - W_i[0])_{\text{ideal}}\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} k(W_i[2] - (W_i[0])_{\text{ideal}}\left(\frac{1}{A_i}\right)
\]
\[ + \sum_{\text{some terms}} kc_1(W_i[1] - W_i[0])\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} kc_2(W_i[2] - W_i[0])\left(\frac{1}{A_i}\right)
\]

Note that summation over some terms means summation over a certain number of terms and not over all the terms 0 to L-1. Also, \( W_i[0] \) can be written as
\[ W_i[0] = k(W_i[0])_{\text{ideal}} + \delta_i \]
where \( \delta_i \) is a constant which varies for each i. Note here that \( (W_i[0])_{\text{ideal}} = -0.25 \) in our case. Finally, the conversion result can be written as
\[ k(\text{ideal result}) + \sum_{\text{all terms}} \delta_i + \sum_{\text{some terms}} kc_1(W_i[1] - W_i[0])\left(\frac{1}{A_i}\right) + \sum_{\text{some terms}} kc_2(W_i[2] - W_i[0])\left(\frac{1}{A_i}\right) \]
The first term \( k(\text{ideal result}) \) adds a gain error. The term \( \sum_{\text{all terms}} \delta_i \) adds a constant offset. The last two terms are the ones which result in the INL and DNL errors. Experimental results indicate that accuracy bootstrapping limits these terms to less than 1 lsb. Note that if the ADC were ideal, these two terms would be zero, \( k \) would be 1 and we would just have an error offset. This offset would be because of the quantization error (reflected by \( W_i[0] \)). Finally, the nearer the nominal weights are to the ideal weights, and the smaller the lower order terms (giving rise to \( \epsilon_1 \) and \( \epsilon_2 \)), the greater the accuracy and linearity.

2.6.2.4 Results

Figure 2.9 shows the transfer curve for a 16 bit ADC with two redundant stages for a total of 18 stages with a 5% error in the component values (Flash, DAC and amplifier in Figure 2.6). Two extra stages are used to reduce the INL and DNL errors below 1 lsb and hence to achieve a higher resolution. For the linear sweep, the x axis represents the \( 2^n \) points possible for an n bit converter and the y axis represents the corresponding normalized digital output (corresponding analog value). Note the non-linearity in the curve before the calibration is performed. Figure 2.10 shows the transfer characteristic for the same ADC with accuracy bootstrapping applied. As can be seen, accuracy bootstrapping linearizes the transfer characteristic. Figures 2.11 and 2.12 show the INL and DNL errors for the same ADC after calibration. For the INL and DNL graphs, the y axis is in terms of the lsb (\( \frac{1}{2^n} \)). For example, at \( \frac{1}{60000} \) of the input scale, the INL is about 0.15 lsb and the DNL about 0.25 lsb. On the whole, the INL and DNL magnitudes are within 1 lsb. A global offset and gain error is present which can easily be corrected.

2.6.2.5 Effect of Multiple Calibrations

Let us again begin by considering stage 0 at the 2nd calibration.

\[
W_0[1] - W_0[0] = \frac{\Delta V_0^{DAC}[1]A_0}{A}(1 + \epsilon_1)
\]

and

\[
W_0[2] - W_0[0] = \frac{\Delta V_0^{DAC}[2]A_0}{A}(1 + \epsilon_2)
\]

The \( \epsilon \) values might have decreased but we know that the weights in this calibration have a gain error of \( k \) and an offset. The output is therefore

\[
(k(\Delta V_0^{DAC}[1]A_0) + \text{offset})k(1 + \epsilon_{\text{new}})
\]
Figure 2.9 Uncalibrated ADC transfer characteristic

Figure 2.10 ADC transfer characteristic after calibration by accuracy bootstrapping
Figure 2.11 INL for the calibrated ADC

Figure 2.12 DNL for the calibrated ADC
which can be represented as $k^2 \Delta V_0^{DAC} (1 + \text{error})$. The gain error is now $k^2$ with an offset error which will also be greater than previous if $k$ is greater than 1 and lesser than previous on the negative side if $k$ is lesser than 1. Therefore, $k$ controls the transfer characteristic for multiple calibrations.

### 2.6.2.6 Multiple Calibrations

If we were to run multiple calibrations without initializing the weights before each calibration, the curve would deviate significantly from the ideal one though it would still be linear. In Figure 2.13, as we can see, if $k$ is less than 1, the curve shifts downward after a certain number of calibrations. Both the slope and offset increase. Thus the curves confirm what we discussed above. A similar effect can be observed when the gain ratio ($k$) is greater than 1. Thus, in order to average out the effect of noise, the best method would be as follows. Before each calibration, we would have to initialize the weights to their nominal values, perform the calibration, initialize, calibrate and so on and take the average after a number of such runs. This method for calibration is described below.

### 2.6.2.7 Calibration Procedure to reduce Noise Effects

Since noise is a random phenomenon, taking a large number of samples and averaging them would reduce their effect [22]. The following technique can be used to average out the noise effects while taking
into account the effect of multiple calibrations on the transfer characteristic and digital weights.

(1)-(5) These five steps are the same as described in section 2.6.1.2.

(6) This is the stage which differs significantly from the previous procedure for calibration. After one iterative calibration cycle through all the stages, if we were to repeat the same procedure, we have shown that the curve goes further away on either side depending on the gain ratio. The increase in linearity is minimal. Therefore, the following procedure would be better. We would store the values obtained for the weights for each stage and we would re-initialize the weights to their nominal values. Then we would repeat steps 1 through 5. We would again store these weights and do this a number of times. The final weights for each stage would be an average of the weights obtained over all the calibration cycles. This is illustrated in the form of a flow chart in Figure 2.14.
2.6.3 Karanicholas and Lee’s Algorithm: A Non-Binary Radix Algorithm

This technique is based on a radix 1.93 and one comparator per stage conversion algorithm [22]. When the residue of a certain stage exceeds the input range of the subsequent stage, this causes the next residue to be even further out of range. Finally the last residue becomes very large, and the quantization error unmanageably large leading to the conversion result being invalid. The problem can be avoided by increasing the input range of each stage, beyond the nominal output range of the previous stage. This guarantees that the residues and thus the quantization error would remain limited. The input range can be increased using a design where \( M > |A| - 1 \). In the previous section, this was done by increasing the number of comparators \( M \). The Karanicholas and Lee algorithm accomplishes this by decreasing the gain \( A \) with respect to a minimal design which has a gain of 2.

Figure 2.16 shows the pipeline with digital calibration. The basic cell is also shown in Figure 2.15. \( V_{in} \) in the input voltage to the cell, \( V_{out} \) the output residue of that cell, \( D_{in} \) the input digital bit used to select a reference voltage and \( D_{out} \) the output digital bit of the stage. \( D \) stands for the output digital word of the converter. The output of the basic cell is as follows

\[
V_{out} = 2V_{in} - V_{ref} \quad \text{if} \quad D = 0
\]

\[
= 2V_{in} + V_{ref} \quad \text{if} \quad D = 1
\]

The gain must be reduced enough so that the residue is contained within the reference boundary. Excessive gain reduction decreases the total number of decision levels available and requires an excessive number of additional stages. A gain of 1.93 was chosen to ensure enough gain reduction so that the residue never exceeds the reference boundary in the worst case when the maximum capacitor mismatch,
Figure 2.16  Pipeline ADC with digital calibration applied to the eleventh stage [2]

Figure 2.17  Digital calibration of higher stages [2]
comparator offset and charge injection error magnitudes are summed together. Note that we would require more than 15 stages for a 15 bit binary converter. This can be derived as follows for a unipolar converter. Since the worst case is an input of 1 Isb, an input of \( \frac{1}{2^n} \) would go through the first \( L-1 \) stages multiplied by 1.93 and the input to the last stage would be \( \frac{1.93^{L-1}}{2^n} \). This has to be greater than 0.5 (the mid range) to be detected in the last stage. Thus the number of stages \( L \) would approximately be \( 1 + (n - 1) \frac{\log(2)}{\log(1.93)} \). If \( n = 15 \), we get \( L = 17 \). The pipelined ADC described in [22] has the first 11 stages with nominal gains set to 1.93 and the last six stages with nominal gains set to 2. The calibration begins with the eleventh stage and continues with the tenth stage, up to the first stage. Figure 2.17 shows the calibration for the eleventh stage. The residue plot is also shown with an exaggerated gain reduction indicated for simplicity of illustration. \( D \) stands for the output digital bit of the cell being calibrated and \( X \) stands for the digital representation of the residue of that stage as determined by the rest of the pipeline. \( Y \) stands for the output code for \( V_{in} \) for that stage. In other words, \( Y \) becomes the \( X \) for the stage preceding the stage being calibrated. The outputs \( D \) and \( X \) are presented to the digital calibration logic system along with calibration constants \( S_1 \) and \( S_2 \) determined for stage 11. The two quantities \( S_1 \) and \( S_2 \) are identified on the residue plot. \( S_1 \) and \( S_2 \) correspond to the quantized representation of \( V_{out} \), or the quantity \( X \), when \( V_{in} = 0 \) with \( D = 0 \) and \( D = 1 \), respectively.

### 2.6.3.1 Calibration Procedure

The calibration algorithm is as follows:

\[
Y = X, \text{ if } D = 0
\]
\[
Y = X + S_1 - S_2, \text{ if } D = 1
\]

where \( D \) is the bit decision, \( X \) is the raw code (uncorrected) and \( Y \) is the transformed code. This transform ensures that the output code \( Y \) with \( V_{in} = 0 \) is the same for \( D = 0 \) and \( D = 1 \), eliminating missing codes. Note that no multiplication is required in this scheme though the stages cannot be simply replicated in a pipelined fashion as in the previous case. Two constants \( S_1 \) and \( S_2 \) are needed per stage. To determine \( S_1 \) the analog input is set to zero and the input bit (\( D_{in} \)) is forced to 0. The quantity \( X \) in this condition is \( S_1 \). In an analogous manner, \( S_2 \) is determined when the input bit is forced to 1. To reduce noise effects, the constants are obtained by averaging a large number of samples.

With the digital calibration of one stage accomplished, the digital calibration of higher level stages can proceed. Figure 2.17 illustrates this. The system within the dashed box has been calibrated so far. The last six stages with gain 2 and the digital calibration logic were used to calibrate stage 11. The
calibrated system within the dashed box is now used as an ADC to measure the residue of stage 10. Calibration constants $S_1$ and $S_2$ for stage 10 are determined in a similar manner as for stage 11. The digital calibration logic system for stage 10 follows the same algorithm as for stage 11. The calibrated system leading with stage 10 can then be used to calibrate stage 9. This process continues up to the first stage. In this way, the entire pipeline ADC is calibrated. Note that if accuracy bootstrapping were applied in this 1 bit per stage, radix 1.93 ADC, we should theoretically get much better results as the first calibration uses the rest of the 17 stages and not the last 6 stages set to gain 2.

2.6.3.2 Mathematical Description of the K & L Algorithm

Since $+V_{ref}$ and $-V_{ref}$ can be different in magnitude, let the two magnitudes be represented by $V_{ref1}$ and $V_{ref2}$. The input voltage can then be written as:

$$V_{in} = \sum_{D=1}^{\text{some}} V_{ref1} \left( \frac{1}{A^i} \right) + \sum_{D=0}^{\text{other}} V_{ref2} \left( \frac{1}{A^i} \right)$$

which can be expressed as

$$\text{ideal result} = \sum_{D=1}^{\text{some}} (V_{ref1} - V_{ref2}) \left( \frac{1}{A^i} \right) + \sum_{D=1}^{\text{all}} V_{ref2} \left( \frac{1}{A^i} \right)$$

Actually, $A^i$ in the above equation should ideally be $A_0A_1...A_{L-1}$ (Note that $i = L - 1$). Considering the first stage,

$$S_1 - S_2 = \frac{V_{ref1} - V_{ref2}}{A^{L-1}}$$

comparing with the ideal term $\frac{V_{ref1} - V_{ref2}}{A^{L-1}}$, the overestimation is $\frac{A^{L-1}}{A_0}$. Call this $k$.

considering the second stage,

$$S_1 - S_2 = \frac{V_{ref1} - V_{ref2}}{A^{L-2}}$$

Therefore this term has two overestimations. One is $\frac{A}{A_0}$ due to the fact that stage 0 is used in measuring the weights. The other one is due to the difference from the ideal weights which is $\frac{A^{L-2}}{A_{L-1}...A_2A_1}$. Therefore the overestimation is $k$ for all the stages. Therefore the result is

$$\sum_i (S_1 - S_2) \left( \frac{1}{A^i} \right) = k \sum_{D=1}^{\text{some}} (V_{ref1} - V_{ref2}) \left( \frac{1}{A^i} \right)$$

$$= k(\text{ideal result} + \text{constant})$$
Thus a global offset and gain results. If the offset and global gain are compensated then a highly accurate transfer characteristic results.

2.6.4 Comparison of Self Calibration Algorithms

Summarizing, two self calibration algorithms were investigated here. The cell in [22] uses 1 comparator per stage and an opamp of gain 1.93 (a purposefully introduced -3.5% error). This results in about 1.8% combined error tolerance in the ADC, DAC and amplifier. The cell in [25] on the other hand converts 1.5 bits per stage with 0.5 bit overlap with the next bit. This is done by introducing an extra comparator in the flash ADC section of the cell. This results in an over-range capability of about 12.5% combined errors in the ADC, DAC and amplifier. Both the algorithms account for some capacitor mismatch, comparator offset, charge injection, finite opamp gain etc. which are lumped as errors in the ADC, DAC and gain values. Based on an investigation of the two algorithms, a 1.5b per stage architecture is decided upon. Table 2.1 summarizes the above comparison.

Table 2.1 Comparison of the two architectures

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Binary radix</th>
<th>Non-binary radix</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bits/stage</td>
<td>1.5</td>
<td>1</td>
</tr>
<tr>
<td>Error Margin</td>
<td>12.5%</td>
<td>1.8%</td>
</tr>
<tr>
<td>No of stages</td>
<td>normal</td>
<td>slightly more</td>
</tr>
<tr>
<td>Digital hardware</td>
<td>simple</td>
<td>more complex</td>
</tr>
</tbody>
</table>

2.6.4.1 Equality of the Two Algorithms

The Karanicholas and Lee algorithm calculates the two calibration constants $S_1$ and $S_2$. Then the result is

$$ Y = X, \text{if } D = 0 $$

$$ Y = X + S_1 - S_2, \text{if } D = 1 $$
Thus, the result would be
\[ \sum_{\text{some terms}} (S_1 - S_2)_{\text{stage } i} \]
which can be represented as
\[ \text{DigitalOutput} = \sum_{\text{some terms}} (S_1 - S_2) \left( \frac{1}{A^1} \right) \]

Thus, the term \((S_1 - S_2) \left( \frac{1}{A^1} \right)\) would be similar to a weighted term in the result of the Soenen Algorithm [25]. Careful analysis of the terms indicates that we calculate \(S_1\) and \(S_2\) for the Soenen algorithm as in Figure 2.18(a) and for the Karanicholas and Lee algorithm as in Figure 2.18(b). But because of the way, the switched capacitor ratioing is implemented in the Karanicholas and Lee architecture, we are in essence doing the same analysis as the Soenen algorithm. The only difference is that the initial measurement in the Karanicholas and Lee architecture is done by the last six stages of gain 2 whereas in accuracy bootstrapping, we use the entire ADC to measure itself iteratively. Note again that if accuracy bootstrapping were applied in this 1 bit per stage, radix 1.93 we should theoretically get much better results as the first calibration uses the rest of the 17 stages and not the last 6 stages set to gain 2. This is presently being investigated by Tapas ray of Iowa State University and the results will be presented in his thesis.

### 2.6.4.2 Weights used in Self Calibration

The Karanicholas and Lee architecture is in effect a radix 1.93 algorithm. Referring to [25] (the binary radix architecture), we obtain that
\[ V_{\text{in}} = V_{\text{DAC}}^{L-1} + \frac{V_{L-2}^{\text{DAC}}}{A_{L-1}} + \frac{V_{L-3}^{\text{DAC}}}{A_{L-1}A_{L-2}} + \frac{V_{0}^{\text{DAC}}}{A_{L-1}...A_{1}} + \frac{V_{\text{ref}}}{A_{L-1}...A_{1}} \]

which is further simplified to

\[ \text{Result} = W_{L-1} + \frac{W_{L-2}}{A} + ... + \frac{W_{0}}{A^{L-1}} \]

where \( W_{l}[c] = \frac{V_{l}^{\text{DAC}}[c]A_{L-1}^{-n}}{A_{L-1}...A_{l+1}} \)

Therefore to apply this to the non-binary architecture, we would have to use a value of \( A = 1.93 \) and not 2 as before. The divide by \( A^L \) function would not be a simple shift as in the case when \( A = 2 \). Thus, if we used a radix of 1.93, the nominal weights would be the same but the arithmetic would be much more complex. On the other hand, if we decided to use a value of \( A = 2 \) (radix 2 or binary system), the nominal values of the weights would not be so simple and would have to be calculated from the formulae. Thus, the extension of the Soen Algorithm for the radix 1.93, 1 comparator per stage architecture is not straightforward and would have to be analyzed in more detail. The numbers involved would not be simple as in the Soen architecture and hence the hardware implementation would be a bit different. As indicated this is under investigation and results will be presented by Tapas Ray of Iowa State University later on.

### 2.7 Conclusions

This chapter dealt with the topic of digital calibration. The origin of various errors that occur in A/D conversion and their effects on the overall transfer characteristic was explained in detail. Various architectures and algorithms to perform digital self calibration were described. A mathematical insight into how these algorithms work was provided. Finally, a comparison of the two algorithms is done. The main conclusion of this chapter is that the technique of accuracy bootstrapping linearizes the transfer characteristic. Hence, a 16 bit converter can be built in present day technology using less accurate components and the technique of accuracy bootstrapping. Also, multiple calibrations tended to shift the transfer curve and a different technique to perform multiple calibrations to reduce the effect of noise was proposed. The next chapter introduces a parallel pipelined architecture to increase speed and discusses a technique to calibrate these multiple pipelines.
3 SIMPAD - A FUNCTIONAL SIMULATOR

3.1 Introduction

With the present trend in the design of A/D converters for high speed/high resolution being dominated by pipeline architectures [12]-[20], there is a great need for a simulator which can be used to model these converters functionally. In the past, there have been some functional simulators for mixed signal and analog sampled data systems like MIDAS [26], but there has not been any available simulator specifically targeted towards pipelined A/D Converters. Most of the designers use packages like Mathematica or Matlab or write their own code. This thesis included an attempt at producing a specialized simulator for single and multiple pipelined A/D Converters. In addition to simulating the functionality of a pipeline, SIMPAD incorporates the capability to simulate the two digital self calibration techniques previously described. Finally, an important specification for converters, especially the ones targeted for communication applications is the Spurious Free Dynamic Range (SFDR). SIMPAD provides the capability to perform single tone and double tone testing for pipelines and to perform FFT's on the resulting data. The core of the accuracy bootstrapping was written by Eric Soenen while he was at Texas A & M university [21]. SIMPAD was developed on this core. A manual for SIMPAD is attached as an appendix.

3.2 Model of a Single Stage of the Pipeline

Since the simulator must account for any kind of architecture similar to the ones described previously, the model used for the single stage is very generalized (refer to Figure 2.1 for a generalized pipeline A/D converter). The number of stages, the number of redundant stages, the gain/stage and the bits/stage are first initialized. The conversion of each stage is set up as a module which targets a specific architecture. This can be replaced by the user to target his/her own model. Three kinds of linear component errors are possible in Figure 3.1. These are errors in the Gain of the Sample/Hold amplifier, errors in the Flash ADC reference levels and errors in the reconstructing DAC. Depending on the user specification,
the distribution of these errors is either uniform or extreme and is random in nature. The gain (A), the DAC level and the ADC levels of each stage \( I \) are stored as

\[
A[I] = A + \delta A_I \\
DAC[i][I] = DAC[i][I] + \delta DAC[i][I] \text{ where } i \text{ is the } \text{ith DAC level} \\
ADC[i][I] = ADC[i][I] + \delta ADC[i][I] \text{ where } i \text{ is the } \text{ith ADC level}
\]

The errors in the above equations are generated randomly. Output referred noise and non-linearity are also added at each stage. Non-linearity is discussed in the next subsection. This is just added to the residue of each stage as

\[
V_{res} = V_{res} + \text{noise} + \text{non-linearity}
\]

The conversion is done by adding the weight of the stage divided by the gain of the stage with the incoming residue to generate the output residue. The output of the flash A/D gives the bits of the present stage.

### 3.2.1 Opamp Model

The gain of the opamp is initialized at the beginning of the program. Furthermore, since the sample/hold is built with an opamp, the model allows for the maximum equivalent output non-linearity \((Ie)\) for each stage to be incorporated. The non-linearity can be modeled as a single bow as in Figure 3.2 or as a double bow as in Figure 3.3. The nonlinearity for a single bow is modeled as

\[
\text{Nonlinearity} = 4 \times le \times V_{res} \times (1 - V_{res})
\]

and for a double bow as

\[
\text{Nonlinearity} = 64/3 \times le \times V_{res} \times (1 - V_{res}) \times (V_{res} - 0.5)
\]
3.3 Program Description

3.3.1 Input and Output Format

The stage parameters can be initialized either at the command line or through an input file. The parameters initialized are the number of stages, the number of redundant stages, the gain per stage and the calibration algorithm used. Then, the errors in the various components of Figure 2.8, i.e. the gain error, the A/D reference errors i.e. the errors in the flash converter of Figure 2.8, the DAC errors, the non-linearity and the output referred noise are initialized as percentages of the full scale. The A/D can then be simulated with or without these errors. The errors are incorporated in a random fashion. The output of the program for different inputs can be output to the screen, to a file in an easily readable format or it can be plotted out.

3.3.2 Routines Used

The various routines used within SIMPAD are for:

1. Ramp or Sinusoidal Input: The A/D can be simulated with both sinusoidal or ramp inputs. The ramp sweeps over the entire range in user specified steps. The sinusoidal sweep is a single sine wave for single tone testing and a sum of two sine waves with frequencies in the ratio of an irrational number for two tone testing.
2. INL and DNL calculation [4]: Two of the most important ADC specifications are the INL and DNL specifications. INL (Integral Non Linearity) is the maximum deviation of the actual transition points in an ADC's transfer characteristic from the straight line drawn between the end points (first and last code transitions). DNL (Differential Non Linearity) is defined as the difference between the ideal bin width of the converter and the actual width of each bin. A bin is defined as the set of input voltages yielding the same output code.

3. Fast Fourier Transform (FFT) [27]: The Fast Fourier Transform can be performed on the output of the files created by the simulator. Plotting capabilities for the FFT are also provided. A more detailed analysis on the testing of A/D converters using the FFT is given in the next chapter.

Figure 3.4 gives an overview of how SIMPAD works. As can be seen, the various modules for the components especially the conversion stage can be replaced by individual models and thus the user can use SIMPAD to target his particular application.

3.4 Conclusions

This chapter described a functional simulator (SIMPAD - SImulation of Multiple Pipelined Analog-to-Digital converters) used to simulate, test and characterize parallel pipelined, digitally calibrated A/D converters. All the results in this thesis were derived using SIMPAD. It was used to design a 16 bit high speed A/D converter and for calibrating multiple pipelines which is described in the next chapter.
Select Self Calibration Algorithm.

Input the stage parameters from a file if so desired.

Define the Stages of a Pipeline.

Initialize error values and incorporate them.

Convert the input Sine or Ramp using the hardware dictated by the input parameters.

Output to the File or the Screen or to a plot.

Perform the FFT.

Output to the File or the Screen or to a plot.

Save The configuration

Perform Digital self Calibration and Initialize the weights of the stages.

Exit

Figure 3.4 Flow chart for SIMPAD
4 A 16 BIT PARALLEL PIPELINED CONVERTER

This chapter gives a description of an architecture for a 16 bit analog-to-digital converter for communication applications. The main goal was to build a 16 bit A/D converter with a spurious free dynamic range (SFDR) near 100 dB operating at high speed (around 20 MHz). The concept of time interleaving to achieve high speed has been used before [28] [29]. Time interleaving is the process by which the input data is sent to different systems periodically and the output of these systems is then combined such that the time interleaving is invisible to the outside. In [30], a parallel pipelined architecture as we propose to use is described, but calibration of the pipes is not done and hence the resolution achieved is only around 8 bits. Multiple parallel pipelines are used in a time interleaved fashion in order to speed up the conversion rate as in Figure 4.1. Each pipeline is individually calibrated by accuracy bootstrapping and the pipelines are calibrated together by a technique termed as global normalization in order to achieve the required accuracy and SFDR.

The architecture and a few terms used in the rest of the chapter are described here. Figure 4.1 shows the architecture proposed. It consists of a front end sample and hold. The input is then time interleaved by an analog demultiplexer to the different pipelines. The output of the different pipelines is then multiplexed and sent to the back end DSP processor which does the calibration and uses the resultant data. The design of the front end sample/hold is the major bottleneck for achieving high speed and is not part of this thesis. A subconverter stands for one of the individual converters, i.e. an individual pipeline in Figure 4.1. Another term used to refer to the individual pipelines is channel.

4.1 Introduction

Parallelism in the signal path is one way to increase conversion speed. In this ADC, the speed critical input S/H and interstage circuits are time interleaved i.e the various paths or ADC's are multiplexed in time [28]. Thus, the multiple pipes are used as if they are effectively a single converter operating at a much higher sampling rate. Within each channel, the speed of each individual pipeline ADC is limited by the settling time of the S/H during hold mode, which is determined by an opamp settling
Figure 4.1 Time interleaved multiple pipeline architecture
Figure 4.2 Two rank architecture

time. Offset, Gain and timing mismatches between the multiple channels give rise to fixed pattern effects which in the frequency domain are manifested as spurious harmonics (see Figure 4.3). For a time-interleaved A/D system consisting of M ADC's in parallel, with overall sampling rate \( F_s \), and each individual channel sampling at rate \( F_s / M \), the undesired frequency components in the output spectrum occur at multiples of \( F_s / M \) in the case of offset mismatch, and as sidebands centered around multiples of \( F_s / M \) in the case of gain and systematic timing mismatches.

The first problem of the interleaving approach is that it requires a very accurate analog Down Sampler (De-Multiplexer) at its input in order to convert a single high speed signal into a certain number say M, low speed analog sampled and held signals. This is due to the time uncertainty (jitter) of the sample and hold (S/H) circuit preceding each sub-converter when switching from sampling to
hold mode, leading to a corresponding uncertainty in the stored value. To eliminate this, a two rank S/H structure is used [31], in which, during the sample to hold transition time occurring at the second S/H, its input signal is held constant by the first S/H (see Figure 4.2).

A second major problem with the time interleaved architecture arises due to mismatches among the sub-converters creating aliasing distortion in the resulting digital output [28]. These mismatches are of two kinds: Offset and Gain mismatches. The effect of these mismatches is illustrated in Figure 4.3.

4.1.1 Offset Mismatch

Channel offset mismatch is due to opamp offset mismatches and charge injection mismatches across the array. It gives rise to fixed pattern noise - in the worst case, for a dc input, each channel produces a different output code. This is more dominant at the lower input level, as the interstage signal alternates between the offset values. In the frequency domain, this is manifested as frequency components or tones at multiples of \( f_s/M \).

4.1.2 Gain Mismatch

Gain errors result in sidebands around multiples of \( f_s/M \). Intuitively this occurs because each individual channel samples the input signal at a rate of \( f_s/M \) causing the input spectrum to be repeated periodically at intervals of \( f_s/M \). When the channels are perfectly matched, these periodic repetitions cancel except at integer multiples of \( f_s/M \). If some mismatch is present, however, these alias components do not cancel completely and appear in the output spectrum. Alternatively, sampling with mismatched channel gains can also be regarded as multiplying (or modulating) the analog input by a periodic discrete time sequence of period \( MT \). The values of the sequence are \( G_1, G_2, G_3 \ldots G_M, G_1, G_2 \) and so on. (\( G \) represents the gain.) For an input sinusoid at frequency \( F_{in} \), this modulation gives rise to spurious

![Figure 4.3 Mismatch effects on the frequency spectrum](image-url)
mismatch products in the output spectrum at frequencies:

\[
\frac{F_s}{M} \pm F_{in}, \quad \frac{2F_s}{M} \pm F_{in}, \quad \ldots, \quad \frac{(M-1)F_s}{M} \pm F_{in}
\]

An interesting point to note here is that according to [29], as the degree of interleaving M increases, the number of aliased components increases, but both the mean and variance of their magnitudes decrease. The following subsections discuss various techniques that are employed to reduce the effects of these mismatches.

4.2 Methods for Calibration of the Paths in a Parallel Pipelined ADC

4.2.1 Hardware Sharing

In this approach, the resources are shared across multiple channels. Use of a pipeline approach for each of the parallel paths allows circuitry, such as bias circuits and resistor strings, to be shared over all channels [30]. Fixed pattern noise effects due to inter-channel mismatches are minimized by appropriate auto-zeroing and by the use of a common resistor string DAC for all the channels. Gain mismatch can be reduced to an extent by sharing the resistor string DAC level generator which also reduces hardware. Note that this approach can be used in collaboration with other digital calibration schemes.

4.2.2 Averaged Parallel Sampling

In video applications, where the sampling rate exceeds the Nyquist frequency, an error averaging technique [32] offers a solution to the inter-channel mismatch problem. The following discussion is for the case when there are two parallel channels as in Figure 4.4. The input signal is sampled by the parallel channels on alternate clock phases, passed through a one tap delay, and summed with the opposite signal thereby averaging the inter-channel mismatches. The parallel signals are then multiplexed to provide a single output at twice the sampling rate. We are in effect using a digital filter of order 1. This can easily be extended to more than two paths. Averaging in this manner reduces the effective input bandwidth and can be tolerated in video applications where oversampling is frequently employed. The averaging operation which is actually a filtering action, places a transmission zero due to the transfer function of the filter at the channel frequency, thus eliminating fixed pattern noise and reducing the sideband effects.
4.2.3 Quadrature Mirror Filtering

The time interleaved A/D converter can be shown to be a special case of a QMF (Quadrature mirror Filter) Bank [33] with the analysis filter, (analog Down Sampler) \( H_k(z) = z^{-k} \) and the synthesis filter (Digital Multiplexer) being \( F_k(z) = z^{-(M-1-k)} \). Basically, in this approach, the input signal is first decomposed into a number of contiguous frequency bands (sub-bands) so that a specific A/D converter (one of the paths) can be assigned to each sub-band signal. Theoretically, the overall resolution depends only on the resolution of the sub-converters used. Although the harmonic distortion due to the mismatches is substantially reduced, the noise generated by the SC circuits, however, places an upper bound of 12 bits [33] and is hence limited to video applications.
4.2.4 Code Error Correction

This technique was discussed in chapter 2 for a multi-step A/D converter [19]. It can conceptually be extended to the multiple pipeline case. The amounts of dislocation of each code from the ideal curve are directly measured during the calibration cycle and stored in memory. The code errors are addressed during the conversion and subtracted digitally from the uncalibrated digital outputs to give the resulting calibrated output. Note that this technique has been used for single ADC’s but can easily be extended to the multiple pipes case.

4.3 Normalization

4.3.1 Introduction

The disadvantage of code error correction is that it would require a large amount of memory to store errors in the case of multiple pipes for a 16 bit ADC. The idea of normalization is to make accuracy bootstrapping do the bulk of the linearization and use the pipes (each of which has already been calibrated) to linearize the resulting curve.

If the transfer curve of each ADC was linear with around 1 lsb of INL and DNL errors, we can calculate the slope and intercept of the transfer characteristic of each of the paths and normalize them with respect to a fixed characteristic. Since the INL and DNL of each of the pipes is linear to an INL error of 1 lsb and DNL error of 1 lsb at the 16 bit level, the resulting normalized curve would also be linear to INL and DNL errors of around 1 lsb (assuming that the slope and intercept are calculated correctly). To calculate the slope and intercept we would have to use a best fit which would require a lot of time and hardware. Instead, we use accuracy bootstrapping to make the individual pipes more accurate (by using a few more redundant stages) and then calculate the slope and intercept by a two point measurement. Since, the individual pipe is linear to around 1 lsb at the 16 bit level, the slope and intercept are accurate in that range. Preliminary results based on this technique indicate that we can achieve 16 bit accuracy using this technique.

In the simulations, we normalize the curve to an ideal slope of 1 and intercept of 0. i.e If,

\[ m_k = \text{slope of transfer characteristic of path } k \]
\[ c_k = \text{intercept of transfer characteristic of path } k \]

then,

\[ \text{Calibrated output} = \frac{(Uncalibrated \ Output - c_k)}{m_k} \]
From the hardware point of view, this can be implemented either on chip (by a set of registers, an adder and a divider) or the calibration and fixing of the measurements can easily be handled by a DSP processor interfacing with this chip (Performing some other function as this would require a very minute effort by the DSP).

### 4.3.2 Results Using Normalization

Figure 4.10 shows the uncalibrated transfer characteristic for a 16 bit ADC with 4 paths each a 16 bit ADC with 2 redundant stages. Because the output oscillates between the transfer characteristics of the different ADC’s it presents itself as a thick line. To make the graph clearer, Figure 4.6 shows the uncalibrated transfer curve for an 8 bit ADC with 2 redundant stages. Figure 4.7 shows the transfer characteristic for the 8 bit converter after global normalization. This figure is more clearer than the 16 bit uncalibrated transfer characteristic in Figure 4.10. Figures 4.8 and 4.9 show the INL and DNL plots for the 8 bit case. The Maximum INL and DNL errors are also shown in Figure 4.10. The other figures show the calibrated linear sweep (Figure 4.11) for the 16 bit case and the corresponding INL (Figure 4.12) and DNL (Figure 4.13) errors. As can be seen, the overall transfer characteristic is linear to INL and DNL errors of around 1 lsb at the 16 bit level.
Figure 4.7  Normalized multiple path 8 bit ADC output

Figure 4.8  INL plot for 8 bit ADC
Figure 4.9  DNL plot for 8 bit ADC

Figure 4.10  Uncalibrated multiple path 16 bit ADC output
Figure 4.11 Calibrated multiple path 16 bit ADC output

Figure 4.12 INL error after calibration for the 16 bit ADC
4.4 FFT Results

SFDR (Spurious Free Dynamic Range) is defined as the ratio in decibels between the magnitude of the fundamental component and the magnitude of the largest harmonic or inter-modulation product. It is an indication of the dynamic operating range of the ADC. As has been discussed earlier, multiple pipes cause undesired tones (referred to in the literature as ping pong spurs) and affect the SFDR of the ADC. Therefore we need to study the Frequency spectrum of the ADC (by means of an FFT) with the pipes included. The following results are for single and two tone testing of the ADC.

4.4.1 Single Tone Testing

In single tone testing, the input is a single sine wave. Up to 95% of the full scale is used in the simulations. The FFT of a single Tone should be a single pulse at $N \times (\text{frequency of sine wave}) / 2\pi$. In order to avoid spectral leakage, we force the input to equal a whole number of periods over the sampling interval. That is,

$$M \cdot T_{in} = N \cdot T_s$$

or

$$\frac{f_{in}}{M} = \frac{f_s}{N}$$
where $N$ is the length of the FFT, $T_{in}$ is the input wave period and $T_s$ the period of sampling. In Figures 4.14 to 4.17, the sampling frequency is set at 20 MHz, and the length of the FFT is set at 1024. An input of approximately 1.5 MHz will be digitized.

$$M = \frac{(1.5 \times 10^6 \times 1024)}{(20 \times 10^6)} = 76.8$$

Therefore we use $M = 77$ and the input frequency is

$$f_{in} = 77 \times 20 \times 10^6 / 1024 = 1,503,906.25 \text{Hz}$$

The process of adjusting the input frequency to a whole number of periods in the sampling window is equivalent in the frequency domain to selecting the spectral line at which the fundamental component will be placed. Also note that the FFT is symmetrical about $N/2$. Hence the x axis in the figures is the frequency normalized to a 1024 point FFT. Note that the y axis is in dB in all these figures. Also 4 channels were used in all the simulations. Figure 4.14 shows the FFT of the input signal. Figure 4.15 shows the FFT of the output of the ADC without any calibration applied whatsoever. Figure 4.16 shows the Output with Calibration applied to each of the individual pipes but with no normalization applied. As can be seen (after careful analysis of Figure 4.16), spurs are observed at

$$\frac{F_s}{2}, \frac{F_s}{4}, \frac{3F_s}{4}$$

and

$$\frac{F_s}{M} \pm F_{in}, \frac{2F_s}{M} \pm F_{in}, \ldots, \frac{(M-1)F_s}{M} \pm F_{in}$$

The number of pipes used in the above simulations were 4. Therefore as has been discussed earlier, the first spurs at multiples of $\frac{F_s}{M}$ are due to offset mismatches and the second spurs are due to Gain mismatches. Finally, Figure 4.17 shows the FFT of the output after Normalization of the transfer curves i.e. after the global gains and offsets are matched. As can be seen, the spurs are reduced considerably.

### 4.4.2 Two Tone Testing

#### 4.4.2.1 With Non-linearity not Included

In two tone testing, two sinusoids, whose periods are not integral multiples of each other are applied to the ADC and the output frequency spectrum is studied. Here the first Sinusoids frequency was the same as the one used for single tone testing. The second sinusoid was chosen to be 2.36328125 MHz. The ratio of the two periods is an irrational number. It was made sure that the input did not go out of range in all these simulations. Also note that the y axis in all the figures is in dB. Figure 4.18 shows
Figure 4.14 FFT of the input

Figure 4.15 FFT of the output with no calibration
pipes individually calibrated - Single tone testing

Figure 4.16 FFT of the output with the channels calibrated individually by accuracy bootstrapping

Globally calibrated or normalized output – single tone testing

Figure 4.17 FFT of the output with global matching or normalization
the FFT of the input. Figure 4.19 shows the output spectrum of the ADC with no calibration applied. Figure 4.20 shows the Spectrum if calibration is applied to the Individual pipes. As can be seen (after careful analysis of Figure 4.20), spurs are observed at

$$\frac{F_s}{M}, \frac{F_s}{2}, \frac{3F_s}{4}$$

and

$$\frac{F_s}{M} \pm F_{in1}, \frac{F_s}{M} \pm F_{in2}, \frac{2F_s}{M} \pm F_{in1}, \frac{2F_s}{M} \pm F_{in2}, \ldots, \frac{(M-1)F_s}{M} \pm F_{in1}, \frac{(M-1)F_s}{M} \pm F_{in2}$$

Note that $F_{in1}$ and $F_{in2}$ stand for the two input sinusoid frequencies. The number of pipes used in the above simulations were 4. Therefore as has been discussed earlier, the first spurs at multiples of $\frac{F_s}{M}$ are due to offset mismatches and the second spurs are due to Gain mismatches.

Finally, Figure 4.21 shows the spectrum with global normalization of the offset and gain mismatches. As can be seen, the SFDR is improved significantly with normalization applied. Note that the above simulations were done assuming no non linearity in the opamps.

4.4.2.2 With Non Linearity Included

In these simulations a non linearity was included in the opamps (see Figures 3.2 and 3.3). Figures 4.22 through 4.25 show the results with an opamp non linearity of 0.001 or 0.1%. As can be seen, the
Figure 4.19  FFT of the output with no calibration (no opamp non linearity)

Figure 4.20  FFT of the output with accuracy bootstrapped channels (no opamp non linearity)
SFDR degrades considerably in this case. Figures 4.26 through 4.29 show the results with an opamp Non Linearity of 0.0001 or 0.01%. The SFDR shows some improvement in this case. Finally Figures 4.30 through 4.33 show the results with an opamp non linearity of 0.00001 or 0.001%. Thus, as can be seen Non linearity of the opamp affects the SFDR considerably and its effect has to be reduced.

4.4.2.3 Results with 3% and 5% Errors

The results presented in the previous sections were all with 1% errors in the component values. The following figures give the same results for 3% and 5% errors in the component values. Figures 4.34 through 4.37 give the results with 3% errors and figs 4.38 through 4.41 give the results with 5% errors. Thus, as can be seen by the various results presented, a spurious free dynamic range of 90 dB seems to be achievable by the use of this technique to perform calibration. Table 4.1 summarizes all the results presented in this section.

4.5 Conclusions

In this chapter, an architecture was proposed for a 16 bit ADC design. High speed was achieved by the concept of time interleaving. To achieve the required accuracy, each individual channel was
Input - Two tone testing

Opamp Non Linearity = 0.1%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.22 FFT of the input (opamp non linearity = 0.1%)

uncalibrated output - Two tone testing

Opamp Non Linearity = 0.1%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.23 FFT of the output with no calibration (opamp non linearity = 0.1%)
Figure 4.24 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.1%)

Figure 4.25 FFT of the output with global matching or normalization (opamp non linearity = 0.1%)
Figure 4.26  FFT of the input (opamp non linearity = 0.01%)

Figure 4.27  FFT of the output with no calibration (opamp non linearity = 0.01%)
Figure 4.28 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.01%)

Figure 4.29 FFT of the output with global matching or normalization (opamp non linearity = 0.01%)
Figure 4.30 FFT of the input (opamp non linearity = 0.001%)

Figure 4.31 FFT of the output with no calibration (opamp non linearity = 0.001%)
pipes individually calibrated – Two tone testing

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.32 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.001%)

pipes Globally calibrated – Two tone testing

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.33 FFT of the output with global matching or normalization (opamp non linearity = 0.001%)
Input - Two tone testing (3% errors)

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.34  FFT of the input (opamp non linearity = 0.001% and 3% component errors)

uncalibrated output – Two tone testing (3% errors)

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.35  FFT of the output with no calibration (opamp non linearity = 0.001% and 3% component errors)
pipes individually calibrated - Two tone testing (3% errors)

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.36 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.001% and 3% component errors)

pipes Globally calibrated - Two tone testing (3% errors)

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.37 FFT of the output with global matching or normalization (opamp non linearity = 0.001% and 3% component errors)
Table 4.1 Summary of all the results

<table>
<thead>
<tr>
<th>Component Errors</th>
<th>Non linearity</th>
<th>SFDR (uncalibrated)</th>
<th>SFDR (paths calibrated)</th>
<th>SFDR (globally calibrated)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1%</td>
<td>0%</td>
<td>40 dB</td>
<td>20 dB</td>
<td>100 dB</td>
</tr>
<tr>
<td>1%</td>
<td>0.1%</td>
<td>30 dB</td>
<td>20 dB</td>
<td>60 dB</td>
</tr>
<tr>
<td>1%</td>
<td>0.01%</td>
<td>35 dB</td>
<td>10 dB</td>
<td>80 dB</td>
</tr>
<tr>
<td>1%</td>
<td>0.001%</td>
<td>40 dB</td>
<td>15 dB</td>
<td>100 dB</td>
</tr>
<tr>
<td>3%</td>
<td>0.001%</td>
<td>25 dB</td>
<td>5 dB</td>
<td>100 dB</td>
</tr>
<tr>
<td>5%</td>
<td>0.001%</td>
<td>15 dB</td>
<td>0 dB</td>
<td>90 dB</td>
</tr>
</tbody>
</table>

calibrated by accuracy bootstrapping. It was also shown by simulations that the spurious harmonics and hence the required accuracy for the time interleaved system was achievable by using a technique termed as global normalization. Hence, we can conclude that a 16 bit ADC at 20 MHz or more is achievable in todays technology. The results have to verified by actually building the various components in silicon. While simulations have been performed for 4 pipelines, it is possible to extend this technique to more pipelines.
Figure 4.38  FFT of the input (opamp non linearity = 0.001% and 5% component errors)

Figure 4.39  FFT of the output with no calibration (opamp non linearity = 0.001% and 5% component errors)
pipes individually calibrated – Two tone testing (5% errors)

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.40 FFT of the output with accuracy bootstrapped channels (opamp non linearity = 0.001% and 5% component errors)

pipes Globally calibrated – Two tone testing (5% errors)

Opamp Non Linearity = 0.001%

X axis: Normalized Frequency
Y axis: Amplitude in dB

Figure 4.41 FFT of the output with global matching or normalization (opamp non linearity = 0.001% and 5% component errors)
5 DESIGN OF COMPONENTS FOR PIPELINED CONVERTERS

This chapter deals with the design of the components used in a pipeline cell. It will serve as a starting point for future work and hence consists of a review of a few architectures targeted towards high speed/high resolution converters. As has been discussed previously, the single pipeline cell consists of a flash converter, a digital-to-analog (DAC) converter, a subtracter and a sample/hold amplifier. Usually, the sample/hold cum DAC cum subtracter is implemented as a single circuit and the flash converter as another circuit.

5.1 Sample/Hold

The reconstructing DAC and the sample/hold is usually implemented as a single circuit. A typical implementation where the feedback capacitor is shared is shown in Figure 5.1.

The operation is as follows:

1.) During the sample phase which is shown in the figure, the input charges the parallel combination of capacitors $C_s$ and $C_F$. Note that both $C_s$ and $C_F$ have the same value. During this time, the flash A/D converter performs its comparison and an appropriate reference voltage is switched onto the terminal termed as $V_{ref}$ in the figure.

![Sample/hold cum multiplying DAC](image-url)
2.) In the second phase called the closed loop amplification phase, the two switches shown in the figure change position. The bottom plate of \( C_F \) is connected to the output and the bottom plate of \( C \), to \( V_{ref} \). Hence, the final output voltage obtained by applying charge conservation is

\[
V_{out} = \frac{(C_s + C_F)V_{in} - C_sV_{ref}}{C_F}
\]

Since \( C_F \) and \( C_s \) are nominally equal to \( C \), the above equation becomes

\[
V_{out} = 2(V_{in} - \frac{1}{2}V_{ref})
\]

Thus, the multiply by 2 stage and sample/hold operation is performed. Note that this is particularly convenient for 1 or 1.5 bits/stage. This architecture has been used extensively [16] [22].

Figure 5.2 shows a differential version of the sample/hold circuit implemented in [30]. Note that the feedback capacitor is not shared in this case. \( V_{CM.in}, V_{CM.out} \) and \( V_{CM.bias} \) are DC bias levels. Note that there is no requirement for the common-mode levels at the input and output of the opamp to be equal. In practice, they are not equal and are chosen and optimized independently. The common mode level at the input \( V_{CM.in} \) is set up by a circuit which replicates the drain source voltage and the gate source voltage of the input transistors of the input stage of the opamp used (can be either folded cascode or normal cascaded version). \( V_{CM.out} \) is chosen to be at the center of the output swing of the amplifier in order to maximize the output swing. The dynamic capacitive common-mode feedback loop at the amplifier output is through the gate voltage of the current source again. Note that this configuration does not require the opamp input and output to be shorted together as is the case in conventional common mode feedback loops.

5.2 Design of the Operational Amplifier Used in the Sample/Hold

The speed of the pipeline is determined by the settling time of the opamp during the "hold" mode. This is directly related to the settling time of the opamp during its closed loop mode. The accuracy of the pipeline is related to the gain of the opamp. Thus an intrinsically fast (high unity gain bandwidth) and high gain opamp is required. Typically, single stage designs cannot achieve accuracy higher than 12 bits even if they are cascoded. Hence to achieve higher accuracy, two stage designs have to be explored. Two architectures which achieve high gain are explained briefly in the following sections. The second architecture achieves high gain with high speed but is more difficult to design because getting a stable operating DC bias is difficult.
Figure 5.2 Differential version of sample/hold amplifier
5.2.1 Two Stage Folded Cascode Architecture

The schematic of a typical internally compensated folded cascode opamp [34] is shown in Figure 5.3. The circuit uses a current folding circuit technique to permit direct connection of the drains of a p-channel differential amplifier to the sources of the cascode devices. This requires two additional transistors (M10 and M11) operating as current sources to bias the first stage. The current sunk by M10 and M11 equals the sum of the currents flowing from the differential amplifier (M1 and M2) and the cascode mirror (M3-M6). Under balanced conditions, the current in M4 is equal to the current in M6. If Vin is increased, then \( I_{D2} \) increases by \( \delta I \) and \( I_{D1} \) decreases by \( \delta I \). These changes reflect themselves into a \( \delta I \) increase in M3 and a \( \delta I \) decrease in M4. The result will be \( 2\delta I \) flowing into the resistance at the drains of M4 and M6.

One of the problems with wide common mode input range opamps is that in the unity-gain configuration for large positive common mode, the output will abruptly spike up to the positive power supply voltage. This problem is removed by M12, M13 and M14. This arrangement causes the biasing current in M10 and M11 to track the biasing current in M7. Thus, when the common mode input limit is exceeded, the amplifier experiences soft limiting rather than an output spike [34].

A detailed small signal model of this circuit is described in [35]. The low-frequency gain of the folded...
Figure 5.4 Single stage folded cascode

cascode opamp is

\[ A_v = \frac{g_{m1}g_{m8}}{G_{o1}G_{o2}} \]

where \( G_{o1} = g_{ds6} \) and \( G_{o2} = g_{ds8} + g_{ds9} \). The dominant pole is found to be \( \frac{G_{o1}G_{o2}}{g_{m6}C_c} \). Note that \( g_m \) is the small signal transconductance and \( g_{ds} \) is the small signal drain to source resistance.

### 5.2.2 Single Stage Folded Cascode Design

To gain an insight into the design of these components, a single stage folded cascode opamp was designed. The process was MOSIS 2\( \mu \) CMOS orbit analog process. Figure 5.4 shows the schematic of this design. Note that it is the same as the first stage of Figure 5.3. The various transistor sizes and other parameters are as follows:

\[
\begin{align*}
\frac{W_1}{L_1} &= \frac{W_2}{L_2} = \frac{180\mu}{4\mu} \\
\frac{W_3}{L_3} &= \frac{W_4}{L_4} = \frac{40\mu}{3\mu} \\
\frac{W_5}{L_5} &= \frac{W_6}{L_6} = \frac{20\mu}{4\mu} \\
\frac{W_7}{L_7} &= \frac{W_{12}}{L_{12}} = \frac{125\mu}{2\mu} \\
\frac{W_8}{L_8} &= \frac{W_9}{L_9} = \frac{34\mu}{2\mu} \\
\frac{W_{13}}{L_{13}} &= \frac{180\mu}{2\mu}
\end{align*}
\]
Table 5.1 Opamp specifications achieved

<table>
<thead>
<tr>
<th>Specification</th>
<th>Simulation result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Loop Gain</td>
<td>~300</td>
</tr>
<tr>
<td>Unity Gain Bandwidth</td>
<td>4.2 MHz at 10 pF load</td>
</tr>
<tr>
<td>Phase Margin</td>
<td>85.3 deg</td>
</tr>
<tr>
<td>Input Common mode Range</td>
<td>1.3-4.2 V</td>
</tr>
<tr>
<td>Output voltage swing</td>
<td>1.3-3.5 V</td>
</tr>
</tbody>
</table>

\[
\frac{W_{10}}{L_{10}} = \frac{W_{11}}{L_{11}} = \frac{W_{14}}{L_{14}} = \frac{40 \mu}{2 \mu} \\
I_{bias} = 110 \mu A, \quad V_{bp2} = 1.75
\]

The opamp specifications achieved in simulation are indicated in Table 5.1.

Figure 5.5 shows the gain bandwidth plot. The first waveform is the input and the second the output of the opamp. Note that the above measurements were with a load capacitance of 10 pF whereas the actual opamp (if used in the pipeline) would not see such a high capacitance load in the pipeline. Hence, the opamp can be expected to operate at much higher speed if the output did not have to drive a pad as is the case here. Also, this version of the opamp was found to take up too much area after layout. Hence, a much smaller opamp would have to be designed.

### 5.2.3 Differential Sample/Hold Design

A typical differential version was implemented exactly as shown in Figure 5.2. The sizes used for the various transistors are given below.

\[
\frac{W_1}{L_1} = \frac{100 \mu}{2 \mu} \quad \frac{W_{1+}}{L_{1+}} = \frac{W_{1-}}{L_{1-}} = \frac{20 \mu}{4 \mu}
\]

The two pass gates at \(V_{in+}\) and \(V_{in-}\) are each 150/2. All the other NMOS transistors are 4/4 and PMOS transistors 10/4. Hence, as can be seen the pass transistors take up a lot of area and have to be optimized.
5.2.4 Architecture based on “Gain-Boost” Principle

The gain-boost technique [36] is based on increasing the cascoding effect of T2 by adding an additional stage as shown in Figure 5.6. This stage reduces the feedback from the output to the drain of the input transistor. Thus, the output impedance of the circuit is increased by the gain of the additional stage, $A_{add}$.

$$R_{out} = (g_{m2}r_o(A_{add} + 1) + 1)r_o + r_{o2}$$

where $g_m$ stands for the small signal transconductance and $r_o$ stands for the small signal output resistance ($r_o = \frac{1}{g_o}$). In this way, the DC gain can be increased several orders of magnitude:

$$A_{tot} = g_{m1}r_o(g_{m2}r_o(A_{add} + 1) + 1)$$

Note that the additional stage itself can be implemented as a cascode stage and thus a very high gain can be achieved. In fact, an opamp has been reported [37] where the additional stage and the normal stage are both implemented as folded cascode stages giving a DC gain of around 90dB and a Unity Gain bandwidth of around 116 MHz. Figure 5.7 shows the frequency response of the cascoded gain stage, the additional stage and the combined “gain-boosted” circuit. A more detailed analysis can be found in [37]. The main problem in the design of this opamp is that it is very difficult to attain a stable DC bias where both T1 and T2 are in the saturation region.
5.3 Design of Comparator Used in Flash ADC

A typical comparator circuit [38] is shown in the Figure 5.8. It consists of a p-channel differential input pair (M1, M2), a CMOS latch circuit, and an S-R latch. The CMOS latch is composed of a n-channel flip-flop (M4, M5) with a pair of n-channel transfer gates (M8, M9) for strobing and an n-channel switch (M12) for resetting, and p-channel precharge transistors (M10, M11). $\phi_1$ and $\phi_2$ are the two non overlapping clocks.

The dynamic operation of this circuit is divided into a reset time interval and a regeneration time interval. The two intervals are approximately between $t_1$ and $t_2$ and between $t_2$ and $t_4$, respectively, as shown in Figure 5.9. During $\phi_2$, the comparator is in the reset mode. Current flows through the closed resetting switch M12, which forces the previous two logic state voltages to be equalized. After the input settles on it’s decision, a voltage proportional to the input voltage difference is established between nodes $a$ and $b$ in the end. This voltage will act as the initial imbalance for the following regeneration time interval. In the meantime, as the n-channel flip-flop is reset, the p-channel one is also reset by the two closed precharge transistors which charge nodes $c$ and $d$ to the positive power supply voltage. As a result, the CMOS latch is set to the astable high-gain mode.

The regeneration is initialized by the opening of switch M12. Since the strobing transistors M8
and M9 isolate the n-channel flip flop from the p-channel flip flop when $\phi 1$ is low, the use of two non overlapping clocks performs the regenerative process in two steps. The first step of regeneration is within the short time slot between $\phi 2$ getting low and phi1 getting high. The second regeneration step starts when phi1 gets high and M8 and M9 are closed. The n-channel flip-flop, together with the p-channel flip-flop, regenerates the voltage differences between nodes $a$ and $b$ and between nodes $c$ and $d$. The voltage difference between node $c$ and node $d$ is soon amplified to a voltage swing nearly equal to the power supply voltages. The following S-R latch is driven to full complementary digital output levels at the end of the regenerative mode and remains in the previous state in the reset mode.

A typical comparator (2$\mu$ MOSIS orbit process) designed by Tapas Ray of Iowa State University was used for the cell. The various transistor sizes are as follows:

$$W_1 = W_2 = 16\mu \quad W_4 = W_5 = 8\mu \quad W_6 = W_7 = 16\mu$$

$$W_8 = W_9 = 4\mu \quad W_{10} = W_{11} = 8\mu \quad W_{13} = 16\mu \quad W_3 = 32\mu$$

The rest of the transistors are minimum sized. Specs Achieved by simulation for this comparator were:

Speed = 20 MHz, Resolution = 10 bits, Input Dynamic Range = 2V
Figure 5.8 Schematic of the comparator

Figure 5.9 Time relation between $\phi_1$ and $\phi_2$
5.4 Conclusions

Once the design of the sample/hold and comparator is completed, the rest of the design basically involves a switching network to implement the flash ADC and the DAC or reference generator. Figure 5.10 shows the final layout of the single cell and other components in MOSIS 2μ technology. At present, the single cell is in the process of being fabricated. Therefore, as has been seen, the key issue involves design of the opamp and the comparator. Thus, the aim of this chapter was to introduce the reader to these design techniques. Future work will have to focus on these key issues and investigate them carefully to come up with a silicon solution in present day CMOS technology.
6 SUMMARY, CONCLUSIONS AND FUTURE WORK

6.1 Scope

The goal of this thesis was to investigate a number of calibration techniques to be used in the design of high performance analog-to-digital converters. The thesis incorporates a considerable amount of literature review in order to serve as a reference. A detailed mathematical investigation of the calibration algorithms is provided in order to give an insight into how these techniques work. An important objective was to develop a software environment to simulate, design and test these techniques in order to serve as a platform for future work. Parallelism (for speed) is incorporated into the architecture by using multiple time interleaved pipelines. A technique to calibrate these pipelines to eliminate spurious harmonics is also described. Finally, a brief review of design techniques for high speed/high resolution is presented. An attempt to implement some components is also presented.

6.2 Summary

Chapter 1 of this thesis provides an introduction to analog-to-digital converters in general. The motivation for this thesis and a general introduction to A/D converters is provided. Definitions used to characterize these converters are explained. Various architectures used to perform data conversion are described briefly. Finally, the organization of the thesis by chapters is presented.

Chapter 2 deals with the topic of digital calibration. The origin of various errors that occur in A/D conversion and their effects on the overall transfer characteristic is explained in detail. A review of calibration, both in the analog and digital domains is provided. Various architectures and algorithms to perform digital self calibration are described. A mathematical insight into how these algorithms work is provided. Finally, a comparison of the two algorithms is done.

Chapter 3 describes a functional simulator that was developed as a part of this work. It begins by describing the models used for the pipeline cell. It also elucidates on the model used for the opamp to incorporate non-linearity. It then describes the input/output format for the software, and the various
routines used to perform characterization and testing. A flow chart for the program is also provided. A manual for this chapter is also provided as an appendix.

Chapter 4 deals with the design of a 16 bit A/D converter. Multiple pipelines are used to achieve the desired speed. The chapter begins with a brief review of similar existing techniques. The individual pipelines are calibrated by accuracy bootstrapping. The chapter then describes a technique called normalization to calibrate the pipelines globally. Single tone and two tone testing to perform characterization by means of a Fast Fourier Transform (FFT) is then described. Various results using the functional simulator are then presented.

Chapter 5 deals with a review of the design of components used in a single cell of the pipeline. Firstly, the design of the sample/hold cum multiplying DAC circuit is described. The design of an opamp for high speed/high resolution is then delved into. Finally, the design of a typical comparator to be used in the flash converter is described. Chapter 6 provides the scope and summarizes the thesis. Finally the contributions of this work and suggested future work are presented.

6.3 Conclusions

The main conclusions are as follows:

1.) The technique of accuracy bootstrapping linearized the transfer characteristic resulting in a global offset and gain. Therefore, in order to calibrate for multiple pipelines, all the transfer characteristics had to be normalized with respect to one reference characteristic.

2.) Multiple calibrations tended to shift the transfer curve and a different technique to perform multiple calibrations to reduce the effect of noise was proposed.

3.) Parallelism had to be incorporated to increase the speed. Each individual pipeline had to be calibrated by accuracy bootstrapping to achieve 16 bit resolution.

4.) The pipelines had to be calibrated to get rid of inter modulation spurs (spurious harmonics) which occur as a result of mismatches between the pipelines. A technique termed as global normalization was used and simulated. An SFDR of 90 dB at 16 bits in today's technology seems achievable using this technique.

5.) Finally, techniques to design high gain/high bandwidth opamp's have to be investigated in order to achieve 16 bit resolution at moderate speeds. In conclusion, 16 bit ADC's operating in speeds in excess of 20 MHz with an SFDR greater than 90 dB seem to be achievable in today's technology.
6.4 Contributions

The main contributions of this work were as follows:

1.) A mathematical insight was provided into two self calibration algorithms. This analysis was useful in understanding how these techniques worked.

2.) The effect of multiple calibrations on the transfer characteristic was also investigated. A mathematical insight was provided and a procedure to perform multiple calibrations to average out the effect of noise was investigated.

3.) A simulation environment for pipelined, time interleaved converters was developed. It incorporated plotting and testing capabilities including linear ramp and FFT tests. An important characteristic of this simulator was its expandability.

4.) Finally, a technique to perform calibration of multiple pipelines in order to get rid of spurious harmonics was developed. Various error magnitudes were simulated an SFDR greater than 90 dB was achieved using this technique in simulations.

6.5 Suggested Future Work

This work could not obviously exhaust all possible research topics that were encountered. In order to develop a practical application in silicon, the topics of chapter 5 must be implemented. The bottleneck for speed in this case is the front end sample/hold amplifier. Another important part would involve the design of a high gain/ high bandwidth opamp to be used in the pipeline since the speed of operation is dictated by the settling time of the opamp in the closed loop and the accuracy by the gain of the opamp. A high speed/high gain opamp has been reported in [37] but area and performance trade-offs will have to be performed to achieve a 16 bit pipeline using this opamp. The design of the comparator would also require a high gain architecture.

Application of accuracy bootstrapping to non-binary architectures is also a topic worth delving into. Using a non-binary radix would reduce the chance of the residue going over range and hence increase resolution though the calibration would not be straight forward. Finally other calibration algorithms involving different topologies of the parallel pipelines instead of just normalization of the transfer curves could also be investigated.
APPENDIX THE MANUAL FOR SIMPAD

(NOTE: It is assumed that you are familiar with the manner in which a pipelined A/D converter works. Familiarity with the accuracy bootstrapping and digital self calibration algorithms is also assumed.)

INTRODUCTION

The program is run at the shell prompt by typing the line

user name> accu

The program then asks you if you want the self calibration to be done by the accuracy bootstrapping algorithm or by the digital self calibration algorithm. See Figure A.1 for the flow chart.

*** Accuracy Bootstrapping (1) or Digital Self-Calibration (2)?

Choosing 1 results in the Default values being shown. Hitting the <RETURN> key at this stage results in the following menu being presented.

M: choose simulation Method (accu or dscp)
D: Define and initialize ADC
I: Input ADC configuration from a file (and pre-calculated random errors if included in the file)
E: set Error limits
R: eRror cancellation algorithm
P: Perturb configuration (add random errors)
A: display ADC configuration
W: display Weights (digital coefficients)
N: calculate ADC Non-linearity
L: Linear sweep - INL / DNL
O: sinusoidal sweep
C: Calibrate configuration using Accuracy Bootstrapping
B: run Batch of different configurations (Monte-Carlo)
T: scale weights using Two-point measurement
U: sUmmary
F: DAVID'S linearity test (may take a while)
G: DAVID'S lin sweep lsb/3 steps
H: DAVID'S random error calibration test
S: Save ADC configuration and errors to a file
Z: Define Multiple(Parallel) Paths
V: Perform the FFT
K: Calibrate the multiple pipes
Q: Quit

Description of the various options

The various options are described below. Note that the program is insensitive to the case of the letter typed in.

M: Choose simulation method (accu or desp)

Choosing this option returns you to the stage where you are asked whether you would like accuracy bootstrapping to be used or digital self calibration. Once the choice is made, hitting the <RETURN> key results in the original menu returning.

D: Define and initialize ADC

This option is to initialize the initial configuration of the ADC. The various parameters to be initialized are:

No of stages

The first parameter to be initialized is the number of stages in the ADC. A stage is a sample and hold amplifier with a ADC, DAC, a Summer and a gain stage.

Delta

This corresponds to the extra stages in the pipeline. These extra stages are for redundancy in order to achieve accuracy. The resolution of the ADC is only up to the no of bits declared as the no of stages.
Select Self Calibration Algorithm.

Input the stage parameters from a file if so desired.

Define the Stages of a Pipeline.

Initialize error values and incorporate them.

Perform Digital self Calibration and Initialize the weights of the stages.

Convert the input Sine or Ramp using the hardware dictated by the input parameters.

Output to the File or the Screen or to a plot.

Perform the FFT

Output to the File or the Screen or to a plot.

Save The configuration

Exit

Figure A.1 Flow chart for SIMPAD
The delta is just for redundancy and error correction.

**Nominal Interstage gain (A)**

This corresponds to the gain of the amplifier which results in the difference being brought out as the residue. Note that the number of bits per stage is $|A| - 1$. Usually it is kept at 2.

At this stage, the initialization is completed and the nominal configuration along with the initial weights are calculated. Note though that the ADC is initialized with no errors. To incorporate errors, the P option is later used.

**I: Input ADC configuration from a file (and pre-calculated random errors if included in the file)**

As the name indicates, the initial ADC configuration along with the pre-calculated errors can be input as a file. This is valid even for multiple pipes.

**E: set Error limits**

As the title indicates, choosing this option allows us to set error limits on the various parameters responsible for ADC non-linearity. These are as follows:

**Maximum Interstage Gain error**

This is the error in the interstage gain $A$. It is entered in percentage i.e if I enter 1 it means 1% gain error. Note that the default values are shown in the brackets and hitting the <RETURN> key without entering anything results in the default value being used.

**Maximum ADC (flash) error**

This is the error in the Flash ADC which converts the analog sampled input signal to its digital equivalent at that stage. This is also entered as a percentage.

**Maximum DAC error**

This is the error in the DAC converter which converts the Flash ADC output back to its analog equivalent in order to be subtracted from the original signal to generate the residue. This is the error which is mainly removed by the self calibration algorithm. Note that this is also input as a percentage.
Error Distribution

This corresponds to the error distribution throughout the stages. u results in the error through all the stages being uniform. That is any error is generated by a random process. Choosing the e option on the other hand results in the errors being extreme in nature i.e. each error is equal in magnitude to either + or - the maximum as set by the error limit.

Simulate recycling ADC, stages identical (y/n)?

The pipelined converter can also be simulated as a cyclic converter with the output of each stage being fed back to itself. It is the same as a non-cyclic converter except that the stages are separated in time rather than spatially.

Max. equivalent output noise of stage

This corresponds to the maximum output noise of the stage.

Number of averages during calibration

In order to reduce the effect of noise, a number of calibrations can be done successively. This option specifies the number of averages to be done. Usually for the accuracy bootstrapping algorithm it is found that 1 calibration is good enough and further calibrations do not result in any significant improvement.

Max. equiv. output non-linearity of stage

This is self explanatory.

Simulate identical non-linearity in each stage (y/n)

This option queries you on whether to simulate each stage with a similar non-linearity or whether to simulate each stage with a different non-linearity. The default is usually set at a similar non-linearity.

P: Perturb configuration (add random errors)

Choosing this option results in the nominal configuration being "perturbed" from its initial condition. Random errors are generated for various ADC parameters and incorporated. The weights of the various stages are then computed.
A: display ADC configuration

This option allows us to view the various ADC parameters at this stage. It first asks you whether to dump the output into a file or onto the screen. The ADC levels of each stage, the incremental DAC levels of each stage, the nominal gain of each stage and the non-linearity are all output.

W: display Weights (digital coefficients)

This option allows us to view the ideal and actual weights of each stage. As above the output can be output to a file or to the screen. The ideal weights (DAC levels) and the actual Weights (say after calibration) of each stage are output.

N: calculate ADC Non-linearity

This option calculates the ADC non-linearity using certain theoretical measures described in the paper by Soenen and Geiger. Refer to the paper for more details.

L: Linear Sweep -INL/DNL

This option inputs a ramp input to the ADC and sweeps across the entire range. Based on two point measurements, it calculates the ADC linearity in terms of the INL and DNL errors. Plotting options are also available.

O: sinusoidal sweep

This option inputs a sinusoidal signal to the ADC and measures the output. Plotting options are also available.

C: Calibrate configuration using Accuracy Bootstrapping

Choosing this option results in the Accuracy Bootstrapping algorithm being implemented for the ADC under consideration. Once this is done, the ADC configuration, Weights, a linear or sinusoidal sweep can be done and the effect of the algorithm on the accuracy and non-linearity of the ADC can be examined.

B: run Batch of different configurations (Monte-Carlo)

This option allows us to run a batch of different configurations allowing us to perform a statistical analysis.
F: DAVID’S linearity test (may take a while)

This option allows us to run a number of configurations allowing us to perform a statistical analysis.

S: Save ADC configuration and errors to a file

As the name indicates, choosing this option allows us to save the present ADC configuration and errors into a file for later use.

Z: Define Multiple (Parallel) Paths

This option allows us to define multiple paths of the ADC. The number of paths is first input. The paths are then perturbed depending on the users choice individually. Then the calibration of paths individually (accuracy bootstrapping) is done again depending on the user. Overall global normalization is done later by choosing K from the main menu.

V: Perform the FFT

This option allows us to perform an fft. The input files to the fft have to be first saved and named. It also allows plotting capabilities.

K: Calibrate the multiple pipes

This option allows us to globally normalize the ADC’s multiple parallel pipes. It is done by two point measurements. Note that it has to be done after choosing the Z option and calibrating each pipe individually by accuracy bootstrapping.

Q: Quit

*** EXITING accu - GOOD BYE ***
BIBLIOGRAPHY


